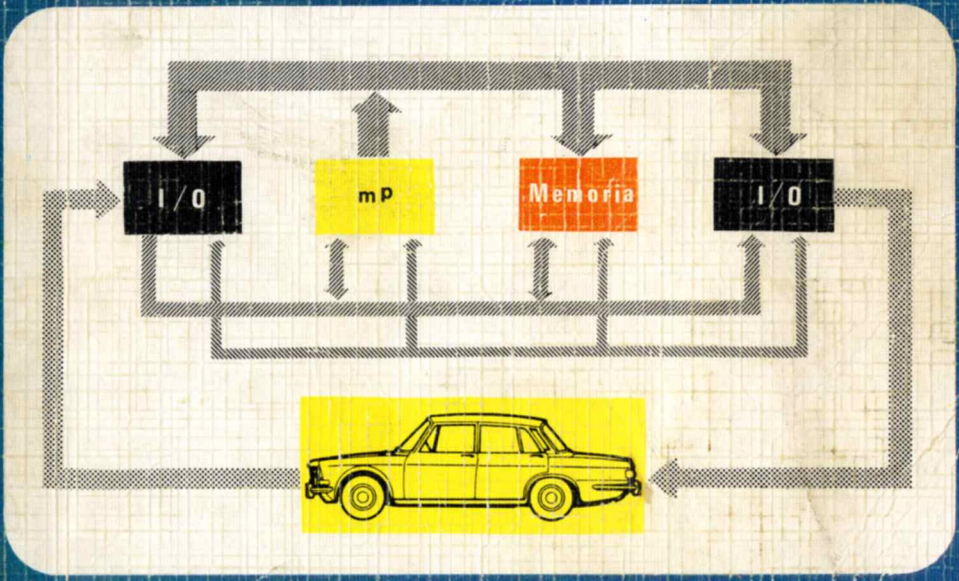
MICRO PROCESADORES

Arquitectura, programación y desarrollo de sistemas

6502
6800
8080

8085
8048
2650



José Mª Angulo Usategui

PARANINFO SA

MICROPROCESADORES

Arquitectura, programación y desarrollo de sistemas

José Mª. Angulo
Ingeniero Industrial
Director de las Academias de Electrónica ANCART
Profesor de "Arquitectura de Ordenadores"
en la Facultad de Informática
de la Universidad de Deusto

MICROPROCESADORES

Arquitectura, programación y desarrollo de sistemas

1981
PARANINFO S.A.
Madrid

© JOSE Mª. ANGULO USATEGUI
1980, Madrid (España)

Reservados los derechos de edición,
reproducción o adaptación.

IMPRESO EN ESPAÑA
PRINTED IN SPAIN

ISBN: 84-283-1087-4

Depósito Legal: M-38652-1980

PARANINFO SA Magallanes, 25 - MADRID (15) (4-2975)

Alco, Artes gráficas. Jaspe, 34. MADRID-26

Indice

Prólogo 11

Capítulo 1: Conceptos preliminares sobre los ordenadores 13

Incidencia de la automatización en la historia del hombre ... 13
Orígenes y fundamentos del ordenador 14
Historia y evolución de los ordenadores 16
Clasificación de los ordenadores 21

Apéndice: Datos y manipulaciones de datos 28

Capítulo 2: Teoría general sobre la arquitectura de los ordenadores digitales 39

Introducción 39
Arquitectura fundamental del ordenador digital 40
Memoria 41
Unidad de control 42
Unidad lógico-aritmética. ALU. Unidad de entradas y salidas . 44
Diagrama por bloques de un ordenador digital 45
Fases de una instrucción 47
Realización de un programa en el ordenador 48

Capítulo 3: Las instrucciones en el ordenador 57

Significado de las instrucciones en el programa almacenado .. 57
Formatos y modos de direccionado de las instrucciones 60
Clasificación de las instrucciones 62
Programación. Diagramas de flujo 63
Juego de instrucciones fundamentales de un ordenador 65
Ejemplos de programación iniciales 68

Capítulo 4: El microprocesador 75

Principios y generadores 75
Arquitectura del microprocesador 77
Arquitectura de un sistema con microprocesador 82
Tratamiento de las entradas y salidas del sistema 86
Búsqueda y ejecución de una instrucción 88
Realización de sistemas de microprocesador 92
Aplicaciones del microprocesador 93

Apéndice: Los elementos triestados en los sistemas con microprocesador 96

Capítulo 5: Arquitectura y constitución de microprocesadores ... 99

Introducción 99
1.ª Parte: Microprocesadores R6500 100
2.ª Parte: Microprocesador Motorola 6800 118
3.ª Parte: Microprocesador 8080 de Intel 127

Apéndice: Microprocesador 2650 de Signetics 164

Capítulo 6: Las instrucciones en los microprocesadores 180

Características y conceptos de las instrucciones 180
Tratamiento general de las instrucciones 182
1.ª Parte: Juego de instrucciones de las CPU, R 6500 185
2.ª Parte: Juego de instrucciones de la CPU MC-6800 202
3.ª Parte: Juego de instrucciones de la CPU 8080 de Intel (8085) 210

Apéndice: Instrucciones del microprocesador Signetics 2650 . 240

Capítulo 7: Familias de circuitos integrados para microprocesadores. Periféricos 253
1.ª Parte: Familia microcomputador para el MC 6800 255
2.ª Parte: Familia microcomputador R-6500 273
3.ª Parte: Familia microcomputador para el 8080 y 8085 del Intel 294

Apéndice: Periférico programable PPI 2655 331

Capítulo 8: Ejemplo y proyectos de carácter didáctico de sistemas con microprocesador 383

Introducción 383
1.º Ejercicio (R6500) 389
2.º Ejercicio (R6500) 390

3.º Ejercicio (R6500) 391
4.º Ejercicio (R6500) 396
5.º Ejercicio (R6500) 400
6.º Ejercicio (R6500) 406
7.º Ejercicio (R6500) 407
Proyectos didácticos resueltos con sistemas con microprocesador 410
1.º Proyecto: Secuencia de encendido de diodos Leds 414
2.º Proyecto: Comprobación de circuitos integrados digitales 424
3.º Proyecto: Posicionado de interruptores 435
4.º Proyecto: Organo electrónico 439
5.º Proyecto: Contador de sucesos y programa 444
6.º Proyecto: Juego para la medida de reflejos 452
7.º Proyecto: Resolución de ecuaciones lógicas con microprocesadores 471
8.º Proyecto: Automatismo resuelto con microprocesadores 475
9.º Proyecto: Control de aparcamiento con microprocesador 483

Apéndice: Ejemplos de programación con el microprocesador 2650 496

A mi hijo Nacho, que nos dejó sus mejores juguetes para automatizarlos con sistemas basados en microprocesadores.

Prólogo

La aparición del microprocesador al comienzo de la década de los 70, ha transformado las técnicas de diseño y control convencionales y constituye una auténtica revolución industrial en nuestros días. Esta revolución, no solamente incluye el aspecto técnico, sino que afectará profundamente a la actual estructura social, en breve espacio de tiempo.

La microelectrónica influye de forma decisiva en la presente y futura tecnología y su rápido avance exige la urgente preparación de expertos, que la puedan aplicar en todos los niveles. El principal motivo que me ha inducido a escribir esta obra ha sido el de ofrecer una visión clara, sencilla y práctica del conjunto de conceptos teóricos y aplicaciones experimentales, necesarios para poder comprender y desarrollar sistemas basados en el microprocesador.

Esta obra se compone de tres partes fundamentales y se puede considerar una continuación del volumen "Electrónica Digital Moderna", cuyo contenido suponemos ya conocido por el lector. Dichas partes son las siguientes:

- Microprocesadores. Hardware y software.
- Aplicaciones industriales de los sistemas con microprocesador.
- Arquitectura de microordenadores y sistemas de desarrollo.

Este primer volumen cubre la primera parte de la obra y se orienta al estudio de la implementación de sistemas básicos de procesamiento de datos, alrededor de un microprocesador. El resto lo cubre el siguiente libro titulado: MICROPROCESADORES II. (Fundamentos, Diseño y Aplicaciones).

Siguiendo la línea que mantengo en otros libros, he orientado éste, intentando exponer los temas en base a una gran sencillez y utilidad. Por esta razón, he desarrollado esta parte de la obra basándome en microprocesadores reales. La elección de los modelos 6800 de Motorola, 8080 y 8085 de INTEL, la familia 6500 de MOS TECHNOLOGY y Rockwell y 2650 de Signetics, ha estado motivada tanto por su popularización a nivel mundial, como a las facilidades de información que de los mismos he obtenido.

Los tres primeros capítulos exponen las bases y fundamentos de la estructura, tecnología y funcionamiento de los computadores digitales en general. El capítulo 4 ofrece las características generales de los microprocesadores. El capítulo 5 se dedica a la arquitectura de los microprocesadores que se han elegido en la obra, mientras que el 6 está destinado al juego de instrucciones de los mismos. El capítulo 7 describe los circuitos integrados de las familias de componentes, que acompañan al microprocesador en la formación de los sistemas. Finalmente, se destina un extenso capítulo a desarrollar ejercicios y diseño de aplicaciones de carácter eminentemente didáctico, cuyo fin es familiarizar al lector con la implementación y programación de sistemas con microprocesador, para que pueda, posteriormente, entender las aplicaciones reales de carácter industrial, ofrecidas en el siguiente volumen.

Para ayudar a la comprensión de la terminología inglesa, tan frecuente en esta técnica, se presenta al final del libro un "Diccionario de palabras inglesas relacionadas con los sistemas con microprocesador", debidamente traducidas.

He intentado en estas líneas conjuntar adecuadamente los conocimientos necesarios de hardware con los de software para poder acceder con base suficiente a un desarrollo práctico. En consecuencia, considero que pueden ser interesantes, tanto a los técnicos electrónicos como a los especialistas en programación.

Agradezco la información recibida de los fabricantes de microprocesadores y la de sus distribuidores en España, destacando a Intel y a su representante Interface, ya que, por mediación de su director, Sr. Forés, he obtenido cuantos datos eran precisos; a Rockwell, representada en España por Comelta; a Signetics y su distribuidor Copresa, de quien destaco las valiosas documentaciones vertidas en su revista Miniwatt; a Motorola y, finalmente, a Hewlett Packard, de quien he utilizado algunos aspectos relacionados con los microprocesadores, expuestos en la documentación de su Laboratorio para Microprocesadores, que comento ampliamente en el libro dedicado a la arquitectura de microcomputadores y sistemas de desarrollo, que puede considerarse la continuación de éste.

Mis alumnos de "Arquitectura de Ordenadores" y de "Microprocesadores y Microcomputadores" del curso de 1980 me ayudaron a corregir, matizar y ordenar algunos aspectos de la obra, de forma relevante los becarios Mikel Echevarría y Fernando León.

Mi dibujante, el Sr. Foronda, ha puesto mucho interés en las figuras, lo que, sin duda, facilitará el acercamiento a los conceptos vertidos en el texto.

CAPITULO 1

Conceptos preliminares sobre los ordenadores

INCIDENCIA DE LA AUTOMATIZACION EN LA HISTORIA DEL HOMBRE

La "necesidad" ha sido lo que ha impulsado al hombre hacia la investigación y el progreso. El arte de la caza, de la agricultura y de la guerra fueron desarrollándose desde el comienzo de la historia de la humanidad para ir cubriendo las necesidades, que en cada época encontró el hombre.

La primera revolución industrial se inicia e identifica con el descubrimiento de la máquina de vapor de Watt en el siglo XVIII. El impacto, que la llegada de las máquinas produjo en la sociedad de aquella época, afectó hasta los cimientos de su misma estructura. Comenzó un desplazamiento masivo de las gentes que vivían en el campo hacia las zonas urbanas, que iniciaban su industrialización. La búsqueda del mayor rendimiento de las máquinas impulsó a sus dueños a la utilización de jornadas de trabajo de 12 y 14 horas, a la suplantación del esfuerzo físico por la máquina, y ofreció posibilidades de trabajos secundarios a mujeres y niños. Pronto se notaron los efectos de las máquinas en la actividad laboral, hasta entonces de tipo artesanal, y miles de personas se quedaron sin empleo.

Tuvieron que pasar bastantes años para que el aumento de la producción y el consumo, produjesen nuevos puestos de trabajo y descendiese el paro. Al mismo tiempo, se estaba produciendo en la sociedad un progreso en la organización del mundo del trabajo y una adecuación al nuevo medio circundante.

Sin embargo, y con ser muy importante la sustitución del esfuerzo físico del hombre por la máquina, otra nueva revolución industrial, de consecuencias mucho más perdurables, se inició en nuestro siglo y en la actualidad contemplamos su desarrollo.

La *automatización* es la palabra clave de la segunda revolución industrial. La máquina, en su primera concepción, sustituía el trabajo muscular humano, pero aún seguía siendo el hombre quien controlaba y gobernaba a la máquina y ésta no podía eliminar el trabajo intelectual. La automatización de la segunda revolución industrial tiene objetivos más profundos: intenta sustituir la acción humana en su vertiente física e intelectual.

El trabajo de cálculo y gestión no fue afectado por la máquina de la primera revolución, antes bien, puede decirse que se incrementó. Sin embargo, los nuevos dispositivos que ofrece la automatización, tratan de simplificar y disminuir el trabajo intelectual humano.

La creciente complejidad de las máquinas y de los procesos de producción y de cálculo azuzó la inteligencia humana para reducir su participación en estos procesos. Los avances espectaculares, alcanzados en los campos de la Electrónica y la Nucleónica, impulsan a otras ciencias y técnicas a un continuo progreso, cuyo objetivo conjunto es el de la automatización. Tanto en los procesos de producción industrial, como en los de diseño, cálculo, investigación y gestión, el ser humano trabaja para disponer de máquinas, que realicen el trabajo con la mínima participación del hombre.

El principal fruto de este esfuerzo investigador de la humanidad, camino de la automatización, ha sido el *ordenador*, máquina fabricada con la más alta tecnología disponible en cada momento, que es capaz de tratar o procesar óptimamente los datos producidos por los dispositivos, que componen tanto los procesos industriales, como los de gestión y controlar las operaciones que de ellos se derivan.

Con lo anteriormente expuesto, puede comprenderse la frase de R. Bendiner, al afirmar que la segunda revolución industrial consistía en la eliminación parcial del trabajo intelectual del hombre.

ORIGENES Y FUNDAMENTOS DEL ORDENADOR

El advenimiento de las máquinas durante la primera revolución industrial trajo consigo un considerable aumento de la producción y del consumo, que motivaron constantes mejoras en los dispositivos de fabricación, en los diseños y cálculos industriales, así como el empleo de técnicas depuradas para el control de ventas y mercados.

En 1904 Fleming, con el descubrimiento de la válvula de vacío, abrió las puertas de la historia de la Electrónica, cuyos formidables avances actúan como motor de impulsión para las restantes ciencias y técnicas.

Al introducir en las máquinas los componentes electrónicos, éstas se fueron haciendo más rápidas, potentes y veloces. En aquella epoca, cada máquina estaba construida y cableada con un determinado propósito o misión. Fue a medidados del siglo XX, cuando Von Newman, continuando la idea de su predecesor Babbage, presentó su máquina de "programa almacenado".

La máquina de Von Newman no estaba diseñada para resolver una aplicación concreta, sino que podía emplearse con muchos fines, previa la conveniente preparación del trabajo secuencial que debía llevar a cabo. Surgió así el concepto de la "memoria", como un elemento capaz de almacenar las instrucciones u órdenes, que debía realizar la máquina ordenadamente, así como los datos y resultados que se alcanzaban. La memoria sustituía al cableado entre componentes, de forma que el trabajo o aplicación de la máquina se podía alterar, cambiando simplemente el programa de instrucciones, contenido en la memoria.

El diagrama general por bloques, en el que se basaba la estructura fundamental de la máquina de Von Newman, se muestra en la figura 1-1.

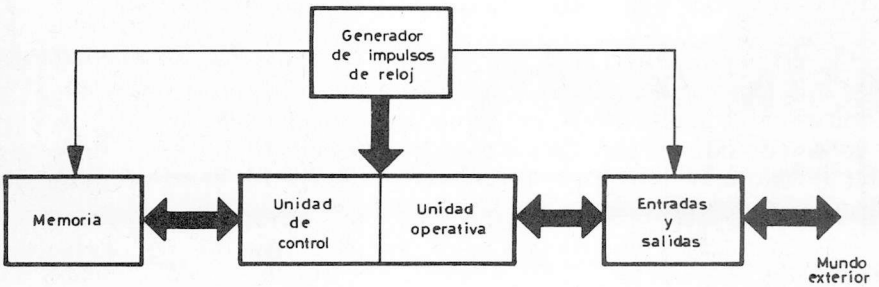


Fig. 1-1.–Estructura básica de la máquina programable de Von Newman.

Con referencia a la figura 1-1, la Unidad de Control, junto con la Operativa, forman la "Unidad Central de Proceso", U.C.P., llamada también C.P.U. (Central Process Unit). La U.C.P. está encargada de ir tomando ordenadamente las instrucciones del programa de trabajo, guardadas en la memoria, e ir decodificando o interpretando dichas instrucciones y generando las señales, que gobiernan el sistema (Unidad de Control), así como ejecutando las operaciones de tipo aritmético o lógico que entrañe la implementación de las instrucciones (Unidad Operativa). El generador de impulsos de reloj sincroniza la realización de las operaciones entre los diversos órganos que intervienen.

En la memoria se almacenan ordenamente las instrucciones, que han de ser ejecutadas por el resto del sistema, para obtener los resultados apetecidos. Sin embargo, la máquina tiene la capacidad de romper la secuencia de las instrucciones y producir saltos en el programa, de acuerdo con los resultados parciales que se estén obteniendo.

Finalmente, la Unidad de Entradas y Salidas se encarga de transformar los datos, provenientes del mundo exterior al código que es capaz de interpretar la máquina, así como de sacar al exterior, en forma útil, los resultados provenientes de la realización del programa de instrucciones.

La notable aportación de Von Newman con el concepto de "máquina programada" constituyó el inicio del planteamiento, que, en la actualidad, se aplica a todo tipo de ordenadores. Sus ideas fundamentales de "programa almacenado" y "rotura de la secuencia de las instrucciones" pretendieron la creación de una máquina muy sofisticada y potente capaz de resolver óptimamente gran variedad de trabajos, debiéndose preparar para cada uno de ellos solamente el programa de instrucciones específico.

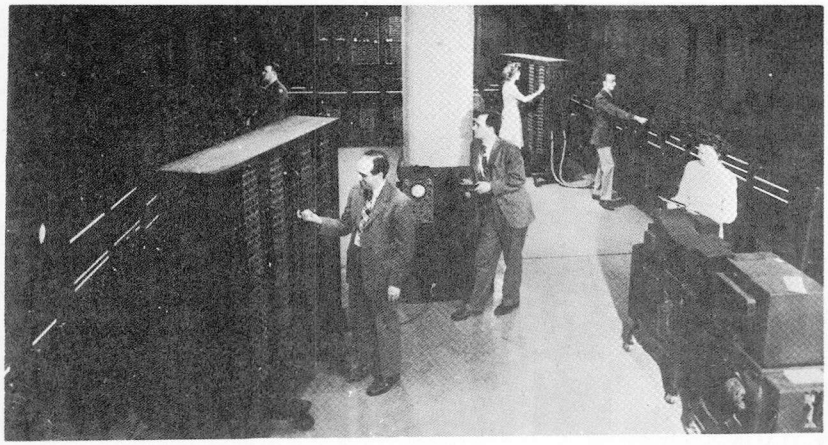
HISTORIA Y EVOLUCION DE LOS ORDENADORES

Las etapas, que han ido recorriendo los ordenadores, han sido las mismas que han ido desarrollando a la Electrónica. Los conceptos que se tenían sobre las máquinas de programa almacenado, sólo se pudieron llevar a cabo, con la aparición de los componentes electrónicos. Así las generaciones de ordenadores surgidas a lo largo del siglo XX, estuvieron marcadas por los descubrimientos de la Electrónica.

1.ª Generación: En 1946 se construyó el primer ordenador electrónico. Sus diseñadores fueron P. Eckertt y J. Mauchly y el ordenador recibió el nombre de ENIAC. Estaba fabricado con unas 18.000 válvulas electrónicas de vacío y ocupaba unos 1.500 m^2. El circuito básico fue el flip-flop, descubierto por Eccles-Jordan. Las memorias utilizaban el sistema de tambores magnéticos y la programación se hacía en lenguaje máquina, mediante cableado externo. Este ordenador sólo tiene un valor histórico, puesto que su coste y dimensiones no permitían su generalización. (Ver figura 1-2.)

Von Newman, continuando el trabajo sobre el Eniac, propuso las bases de los futuros ordenadores: utilización de la aritmética binaria, memoria almacenada y estructura paralela de la máquina. Colaboró en la construcción del computador IAS, que no se terminó hasta 1952.

Fue en 1951 cuando se comercializó el primer ordenador: el UNIVAC I; después, en 1962, IBM presentó el modelo 604 y en 1953 el modelo



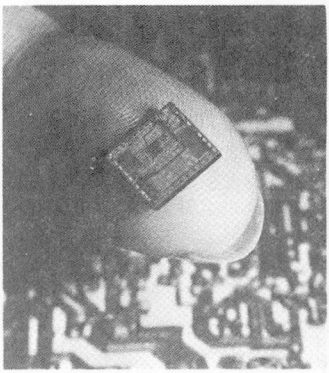


Fig. 1-2.–La fotografía superior muestra el computador ENIAC, construido en 1946 (Cortesía de UPI). En la parte inferior izquierda, una calculadora de mesa, programable, que fabricada con transistores en 1968, es más potente que el ENIAC. Finalmente, abajo, a la derecha, se muestra un chip Microcomputador, que, en un cuadrado de menos de 1 cm de lado, contiene a todos los elementos electrónicos, que configuran un pequeño computador (Cortesía de INTEL).

701, que ocupaba dos grandes salas y costaba alrededor de 1.000.000 $. Por fin, en 1955, IBM lanza su modelo 650, altamente operativo y destinado a fines científicos, que se popularizó en todo el mundo.

2.ª Generación: Como consecuencia del descubrimiento del transistor en 1948, se inicia la segunda generación de ordenadores, en los que la válvula electrónica se sustituye por el transistor, reduciendo considerablemente el tamaño, la fiabilidad y el consumo de potencia. Aunque en

1956 existieron ordenadores con transistores, su aparición en el mercado mundial fue poco antes de 1960. (Ver la figura 1-2.)

Como elementos de soporte de las memorias, además de las cintas perforadas, se introdujeron las ferritas, los discos y las cintas magnéticas. Como lenguajes de programación, se comenzó a emplear un Fortran de tipo elemental.

3.ª Generación: Todos los logros de la Electrónica se aplicaron inmediatamente en la construcción de ordenadores y, así, en 1960, con el descubrimiento del circuito integrado, comienza una era gloriosa en la historia del computador.

El circuito integrado es un simple chip de silicio, de muy pocos milímetros cuadrados, sobre el que se forman e interconexionan transistores, diodos, resistencias y condensadores para dar lugar a circuitos, que implementan funciones completas. Al aplicarse al computador esta nueva tecnología, pasó a ser mucho más pequeño, más fiable y más capaz, con un consumo inferior de energía.

Dentro de esta generación se enmarca la construcción de miniordenadores, que se comentarán posteriormente.

Puntos importantes a destacar en esta generación son:

1) La memoria alcanza capacidades superiores a los 0,5 Megabytes. Asimismo comienza el empleo de memorias de semiconductores.
2) La velocidad de lectura y escritura de datos aumenta mucho. El tiempo que dura un ciclo de una instrucción se mide en nanosegundos (10^{-9} seg.).
3) Dada la potencia y la velocidad de los ordenadores de esta generación, se procesan varios programas simultáneamente, dando lugar a la "multiprogramación". Asimismo, el trabajo del ordenador es en "tiempo real", que consiste en procesar la información y obtener los resultados en el mismo instante en que se van produciendo los primeros.
4) Se utilizan en programación el Fortran, el Cobol y sistemas operativos.

El empleo de circuitos integrados y la aparición de los minis, que, manteniendo la filosofía de los computadores, eran más pequeños y baratos, hacen que los ordenadores correspondientes a esta generación sean los más extendidos en el mundo. La serie 360 de IBM y los sistemas Univac 90/30 son muy representativos a nivel mundial.

En la figura 1-3 se muestra una foto del sistema computador HP 300 de Hewlett Packard, en el que se incluye, además del procesador central

Fig. 1-3.–Sistema Computador HP-300 de Hewlett Packard.

con su unidad de comunicación, una impresora, dos terminales de aplicación y una unidad de discos. El sistema puede soportar hasta 16 terminales, dos impresoras y una capacidad de almacenamiento de discos de 260 Megabytes.

4.ª Generación: Hacia 1971 la aplicación de los últimos desarrollos de la microelectrónica al campo del ordenador, permitió a la casa INTEL fabricar en un solo chip de circuito integrado, LSI (Alta escala de Integración), la Unidad Central de Proceso de un ordenador usando la tecnología MOS. A dicho chip se le llama "microprocesador". A partir de este momento, ésta y otras firmas se volcaron en la investigación y desarrollo de los microprocesadores, inundando el mercado con modelos cada vez más potentes y veloces.

La utilización del microprocesador permite implementar sobre una tarjeta de circuito impreso todo un sistema de ordenador, que recibe el nombre de "microcomputador". En la figura 1-4 se muestra una fotografía del microcomputador AIM-65 de la casa Rockwell, basado en el microprocesador R 6.502, que dispone de un teclado completo, presentador de 20 caracteres e impresora rápida y se puede conectar a 2 magnetofones normales para grabación y reproducción de programas.

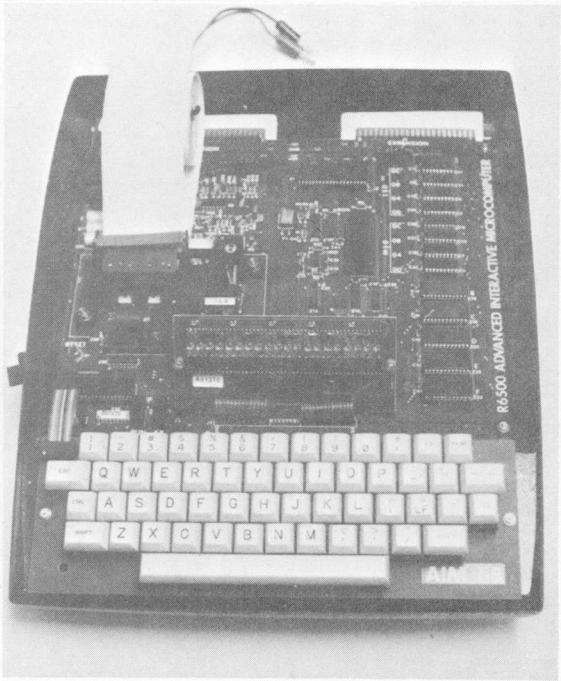


Fig. 1-4.–Fotografía del microcomputador AIM-65.

En la cuarta generación se emplea con profusión las memorias a semiconductores integradas, de capacidades de almacenamiento cada vez mayores, al mismo tiempo que aparecen nuevos lenguajes conversacionales, como el Basic, el Algol, el PL/1, etc.

La aparición del microprocesador ha permitido extender la aplicación del proceso de datos, no sólo al campo concreto de los ordenadores, en el que se ha conseguido la construcción del computador personal o doméstico, como el mostrado en la figura 1-5, sino que otros campos tan diversos como la automoción, el tráfico de ascensores, las máquinas herramientas, etc., se han visto potenciados extraordinariamente con la utilización del microprocesador. Lo anteriormente comentado abre unas espectativas inmediatas de empleo de sistemas de procesamiento en todas las áreas y a todos los niveles.

La reducción de volumen y precio, que proporciona el microprocesador, permiten su aplicación masiva y la necesidad de considerarle, dentro de la Electrónica Digital, como otro componente de características muy sofisticadas.

Fig. 1-5.–Microcomputador personal TRS-80, basado en el microprocesador Z-80 de Zilog.

5.ª Generación: La consecución de un microordenador en un solo chip, permite atisbar en un futuro próximo, no sólo una disminución de volumen y precio en los sistemas ordenadores, sino una aplicación masiva de los mismos con posibilidades insospechadas. (Ver figura 1-2.)

En la figura 1-6 se muestra la fotografía de un chip que contiene todo un sistema completo de ordenador: UCP, memoria con el programa de aplicación grabado, reloj y unidades de entrada y salida.

CLASIFICACION DE LOS ORDENADORES

Atendiendo a la constitución interna de la Unidad Central de Proceso, que se considera la parte fundamental en el tratamiento de la información, los ordenadores pueden clasificarse en: Analógicos y Digitales.

Los ordenadores analógicos están implementados con circuitos electrónicos, capaces de tratar magnitudes variables. Los Amplificadores Operacionales, por su capacidad de realizar operaciones matemáticas alterando

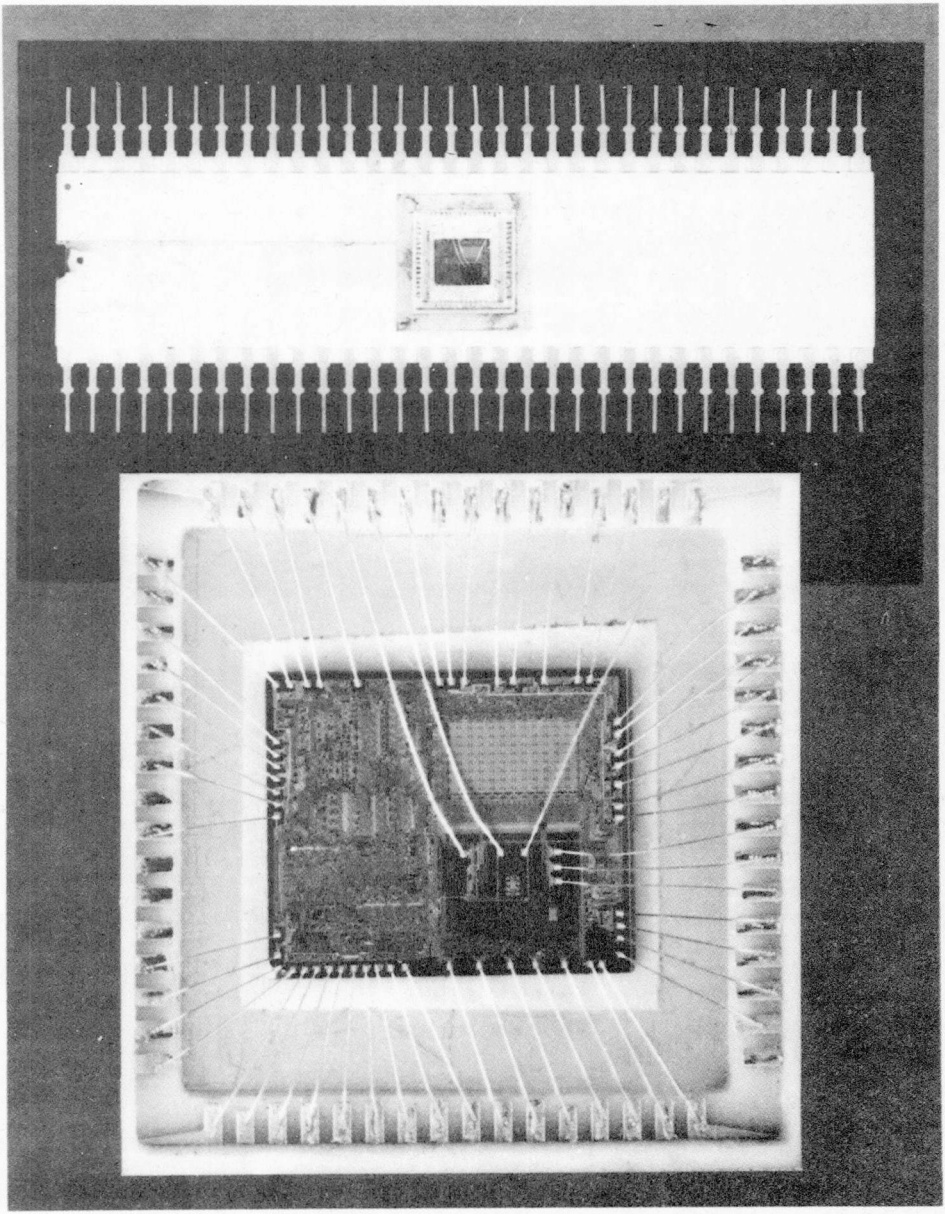


Fig. 1-6.–La foto muestra un chip de Rockwell, que contiene la UCP, memoria, reloj y unidades de entradas y salidas.

su realimentación, son los elementos más usados en este tipo de ordenadores. Se caracterizan por su velocidad en cálculos matemáticos complejos y por su facilidad en simular procesos reales.

Sin embargo, la causa del progresivo abandono de esta tecnología en la fabricación de ordenadores se debe a la insuficiente precisión en los resultados y a su concepción orientada a un campo de aplicaciones concreto y reducido.

Los ordenadores digitales están construidos básicamente por componentes y circuitos de la Electrónica Digital, caracterizada por tratar únicamente dos niveles de magnitud, llamados niveles 0 y 1. Puede alcanzar la precisión que se desee y su estructura es más sencilla y fiable que la que emplea componentes analógicos.

El continuo desarrollo de la Electrónica Digital, al ser absorbido de inmediato por los fabricantes de ordenadores, permite ofrecer continuamente sistemas más veloces, más potentes, menos voluminosos y más baratos. Esto supone que, en pocos años, sistemas que fueron líderes en la optimización de sus características, queden obsoletos.

También hay un tipo de ordenador, llamado "híbrido", en el que se emplea la técnica digital en la UCP y restantes bloques, que requieren mucha precisión, como las memorias, empleando circuitos analógicos en las entradas y salidas de información, bien para adecuar la codificación de los datos entre la UCP y los periféricos, o bien, para adaptar las velocidades de transmisión de datos entre el ordenador y el mundo exterior.

Se pueden establecer numerosas clasificaciones de los ordenadores, atendiendo a cualquier característica notable por la cual quede definido, como pueden ser:

Volumen.
Precio.
Capacidad de memoria.
Potencia de sus instrucciones.
Lenguajes aplicables.
Velocidad.
Aplicaciones y/o periféricos adaptables.

Todas las posibles clasificaciones, teniendo en cuenta características relevantes o un promedio de las mismas, siempre serán subjetivas y variables con el tiempo, pues el avance de la tecnología electrónica es tal, que el volumen, la velocidad, el precio, etc. del ordenador varía rápidamente. Así por ejemplo, el gigantesco ordenador que se construyó en 1946, el ENIAC, con 18.000 válvulas, varias toneladas de peso y ocupando una superficie de 1.500 m^2, puede implementarse en la actualidad sobre una

simple tarjeta de circuito impreso, con sólo 3 ó 4 chips y es factible que un fabricante implemente al gigante en un solo chip.

Por lo tanto y teniendo en cuenta el momento actual, se puede hacer una clasificación generalizada de los ordenadores, atendiendo a un conjunto de características fundamentales; por ejemplo:

1.°) Sistemas con microprocesador.
2.°) Microordenadores.
3.°) Miniordenadores.
4.°) Ordenadores propiamente dichos, en cuyo grupo cabe distinguir tres tipos: grandes medianos y pequeños.

Los dos primeros tipos de ordenadores entran de lleno dentro de la cuarta generación, pues utilizan un sistema ordenador, organizado alrededor de una UCP, fabricada en un solo chip (microprocesador). En general, los restantes bloques del sistema lo forman unos pocos circuitos integrados más.

Un sistema con microprocesador reune todas las características de un ordenador, pero con la potencia mínima para el tratamiento de los datos. Un sistema con microprocesador se diseña siempre para una aplicación concreta y resulta la forma más sencilla y económica de resolver los problemas de tratamiento de la información, mediante un sistema tipo informático. En la figura 1-7, se muestra una fotografía de diversas aplicaciones de sistemas con microprocesador: Control de la sintonía de una radio, aparatos para la navegación aérea, bicicleta de gimnasia con programas personalizados, terminal para el control cromatográfico y unidad portátil para transacciones de información, vía teléfono.

Aunque a veces se confunde el término "sistema con microprocesador" con el de "microcomputador, he preferido distinguir entre ambas denominaciones, dada la importancia que se da en este libro a las mismas. Los microcomputadores son máquinas que procesan los datos, mediante una UCP, consistente en un microprocesador y según un programa de instrucciones almacenado en la memoria. La diferencia entre "sistema con microprocesador" y "microcomputador" radica en que, en el primero, el programa de instrucciones es fijo y el diseño físico del circuito electrónico se ha realizado para resolver una determinada aplicación, por ejemplo, el control del encendido del motor de un automóvil. El microcomputador es de propósito general, es decir, no contiene un programa fijo de instrucciones, pudiéndose cargar en él cualquier programa que permita sus especificaciones.

Tanto los sistemas con microprocesador como los microcomputadores tendrán las características distintivas del microprocesador en que se basan, siendo en la actualidad las mas representativas:

Fig. 1-7.–Diversas aplicaciones de sistemas con microprocesador.

1) Velocidad del ciclo de instrucción. Del orden de 0,5 microsegundos.
2) Tratamiento de palabras de varios bits. Suelen ser las palabras de 4,8 y hasta 16 bits. Se anuncian los de 32 bits.
3) Capacidad de direccionamiento de memoria. Del orden de 64 K bytes.
4) Diferentes modos de direccionamiento de la memoria.
5) Interpretación de un juego de instrucciones de 50 a 100, cada una con diversos direccionamientos.
6) Precio de origen del chip, inferior a las 1.000 ptas.
7) Esfuerzo constante por elevar los lenguajes de programación.
8) Tendencia a la fabricación de todo el sistema en un solo chip.

Los miniordenadores son máquinas similares a los grandes ordenadores que surgieron en la tercera generación, hacia el año 1966 y que están muy extendidos por todo el mundo. Sus propiedades más destacables son:

a) Precio bastante inferior al de los ordenadores. Como cifra orientativa, se puede citar la de 2.000.000 ptas., para el "sistema básico".
b) Velocidad algo mayor que la de los microcomputadores.
c) Tratamiento de palabras de 16 y 24 bits.
d) Software muy completo.
e) Lenguajes de programación de alto nivel, como el BASIC.
f) Capacidad de resolución de un solo problema en cada momento.
g) Elevada capacidad de memoria. Pueden soportar unos 64 K de memoria central y hasta 30 Megabytes en memoria periférica, a base de discos.

Finalmente, el grupo de los ordenadores, propiamente dichos, se distingue por poseer unas características muy superiores respecto a las categorías anteriores, dentro de cualquiera de sus tres versiones: grandes, pequeños o medianos. Se ponen de relieve las siguientes características:

1) Elevado precio de coste y volumen, superior a las restantes categorías.
2) Capacidad de memoria prácticamente ilimitada.
3) Velocidad de trabajo enorme, inferior a los 30 ns.
4) Dada la gran velocidad y capacidad, el ordenador trabaja en tiempo compartido, para varios usuarios a la vez.
5) Utilización de sistemas conversacionales entre el ordenador y el usuario, que permiten a éste controlar todas las fases de desarrollo del procesamiento.
6) Posibilidad de controlar con un gran ordenador a otros muchos de categoría inferior. Esto tiende a ser el principal destino de este tipo de máquinas.

7) Lenguaje de programación de alto nivel, orientados a resolver problemas en diferentes áreas, como el Fortran y el Cobol.

Otra clasificación interesante, que cabe hacer a los ordenadores, es la que se refiere a su aplicación y que se resume así:

A) Destinados a trabajos científicos. Con posibilidades muy potentes y desarrolladas de cálculo matemático.
B) Destinados a la gestión administrativa. Controlan y procesan datos de grandes ficheros para la consecución de nóminas, stocks, estadísticas sobre producción y venta, etc.
C) Destinados al control de procesos industriales. Sirven para gobernar procesos de tipo industrial, conectados directamente al mismo y en tiempo real.
D) Destinados al uso personal, como consecuencia de la aparición de los microprocesadores y la notable reducción de volumen y precio.

Con objeto de dar una visión actual del desarrollo y aplicación de la Electrónica Digital al campo de los ordenadores, se ofrece una amplia exposición de los dos primeros grupos de ordenadores, correspondientes a la cuarta y quinta generación: los sistemas con microprocesador y los microcomputadores. Pero el fundamento de todo sistema de ordenador, independientemente de su potencia o categoría, es siempre el mismo y puede aplicarse a todos ellos.

APENDICE

DATOS Y MANIPULACIONES DE DATOS

INTRODUCCION

El ordenador es una máquina capaz de procesar los datos que se le introducen, de acuerdo con un programa de instrucciones, que ya contiene. Los datos son el principal elemento de trabajo del ordenador y su tratamiento, el fin al que se destinan. Por este motivo, al iniciar el estudio del ordenador, conviene definir claramente su significado, especialmente, teniendo en cuenta lo poco habituados que están los técnicos especialistas en Electrónica Digital a su uso.

La palabra "dato", en el sentido más amplio, se define como cualquier información, que se utiliza o interviene en la realización de una decisión. El "proceso de datos" se refiere al empleo de los datos en un cálculo, su organización, o su participación en otras operaciones, que se realizan secuencialmente de forma manual o automática. La "comunicación de datos" consiste en cualquier movimiento o traslado, que se efectúe desde una fuente hasta su destino. La transmisión de datos se realiza con la participación de un receptor, un transmisor y un canal de comunicación.

En Electrónica Digital los datos se definen como cualquier información, que está representada por un conjunto de bits lógicos 1 ó 0. El proceso de datos involucra la manipulación de los bits 1 y 0, para lo que generalmente, se requiere la codificación de dicha información.

El ordenador constituye el mejor ejemplo de manipulación y transmisión de datos. En efecto, el término "proceso automático de datos" se originó con los computadores. Todos los circuitos lógicos, que poseen los computadores, sólo tienen como fin adquirir, procesar o comunicar datos. Las puertas y otros elementos lógicos, como los flip-flop, registros de desplazamiento, contadores, etc., se utilizan para el tratamiento o almacenamiento de los bits 1 y 0, lo que exige amplias redes para la transferencia de datos desde los computadores y hacia ellos, así como entre los centros de almacenamiento de datos o memorias.

El objetivo de este apéndice consiste en exponer los conceptos generales, que son comunes a los computadores y otros dispositivos lógicos, sobre los códigos, que se basan en los bits 1 y 0, usados para la representación de letras del alfabeto, números y símbolos, así como de los modelos de grupos de bits empleados en la intercomunicación entre circuitos lógicos.

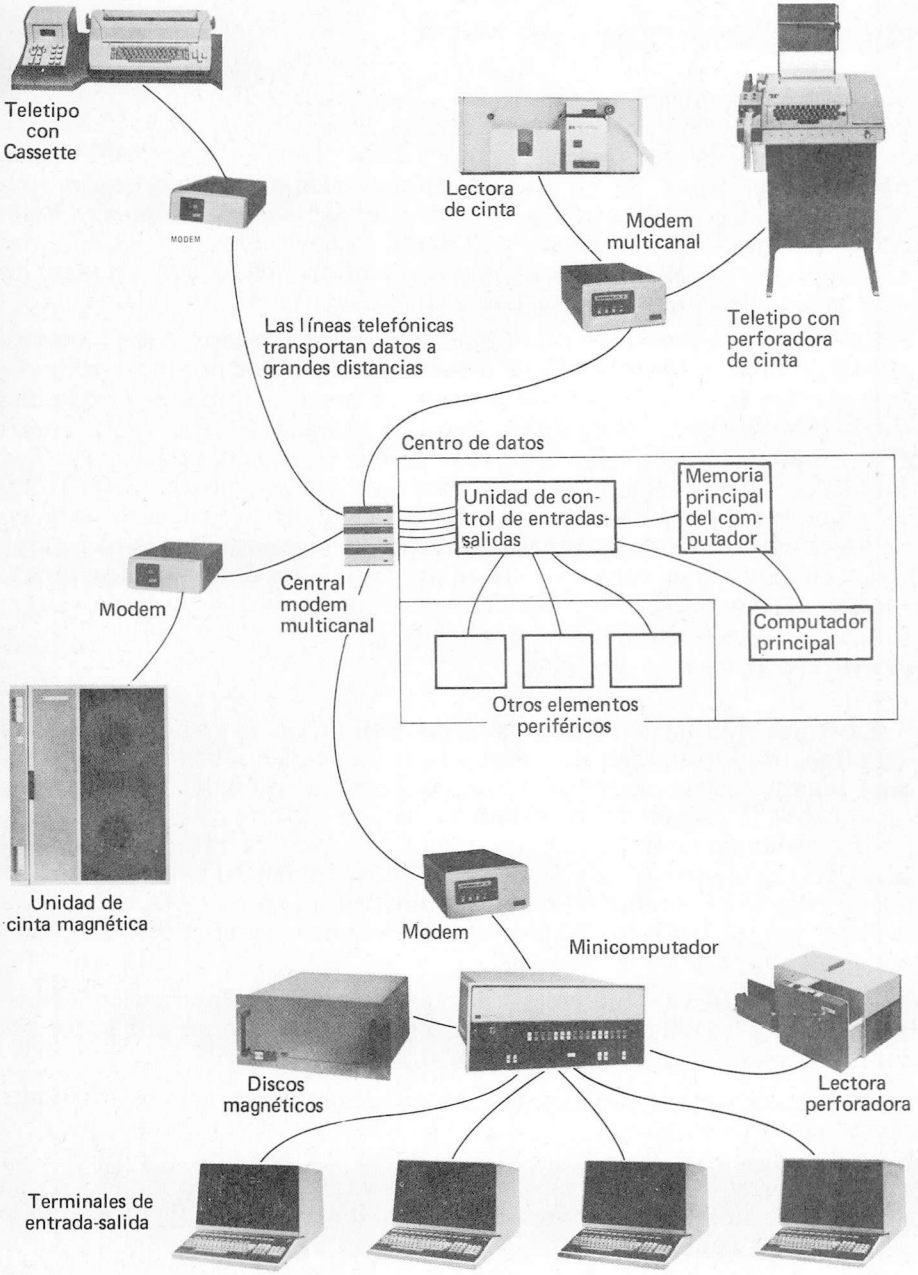


Fig. 1-8.–Sistema de un computador para tratamiento y transmisión de datos.

REDES DE TRANSMISION DE DATOS

Para la comunicación de datos los computadores utilizan las líneas del teléfono, redes independientes e incluso otros medios, como el sistema de televisión. Las redes enlazan grandes distancias: de forma inalámbrica, con cable de conexión, vía línea telefónica, con enlaces de transmisión por microondas, e incluso, vía satélite. Aunque la red telefónica es todavía el sistema de comunicación más usado en el mundo, muchas redes independientes, de carácter no militar, están creciendo con rapidez, en especial aquéllas que están asociadas a los computadores.

La figura 1-8, muestra un determinado número de terminales remotos, intercomunicados a un centro de tratamiento de datos por una red de comunicación. El centro de datos incluye un gran computador con su memoria principal, una unidad de control de entradas y salidas, que regula el movimiento de datos, así como memorias y terminales especiales. Si la conexión entre el centro de datos y los terminales se realiza vía telefónica, como ésta lleva la información en forma analógica, se necesita convertir los dígitos binarios en señales analógicas y viceversa, por lo que precisa un módem. El módem consiste en un modulador-demodulador, que se encarga de la codificación y posterior decodificación.

CODIGOS ALFANUMERICOS

La típica transmisión de datos consiste en una serie de números, letras del alfabeto y otros símbolos, todos los cuales deben representarse en código binario, tal como el 8421. Los dos códigos estándar más comunes son el ASCII (American Standard Code For Information Interchange, Código estándar americano para intercambio de la información) y el EBCDIC (Extended Binary Code Decimal Interchange Code, Código ampliado para intercambio de decimal a binario). El códico ASCII fue desarrollado por el Instituto Americano de Normalización y está aceptado por la mayor parte de la industria.

El código EBCDIC fue creado por la Corporación "International Busines Machines", IBM, y es el empleado fundamentalmente por todos sus computadores.

Ambos códigos, el ASCII y el EBCDIC, incluyen caracteres alfanuméricos, signos de puntuación y otros símbolos, así como órdenes para conocer el estado de las señales. Las órdenes o comandos permiten a un computador y sus periféricos comunicar caracteres, informando de su situación en una página impresa. La figura 1-10 muestra algunos comandos típicos y caracteres codificados. Los datos impresos, generados por la impresora del computador, se muestran en la figura 1-11.

Fig. 1-9.—Algunos típicos terminales de un computador. (Cortesía de Hewlett Packard.)

CARACTER O COMANDO	CODIGO ASCII	CODIGO EBCDIC
A	1000001	10000011
B	1000010	01000011
Z	1011010	10010111
;	0111011	01111000
BACKSPACE (BS)	0001000	01101000
FIN DE LA TRANSMISION	0000100	11101100

Fig. 1-10.–Codificación de caracteres y comandos.

Cuando un operador pulsa la tecla de un aparato de entrada de datos, como puede ser un teletipo, el carácter se codifica en lógica binaria para el computador. Los caracteres codificados, al introducirse al computador, se procesan. Los resultados del proceso pueden ser devueltos al mismo terminal (teletipo) en forma codificada y éste grabar los caracteres adecuados, previa decodificación de los datos que recibe.

Además del terminal de teletipo, existen otros, que codifican los caracteres, bien perforando tarjetas de papel, bien en cinta perforada, en lugar

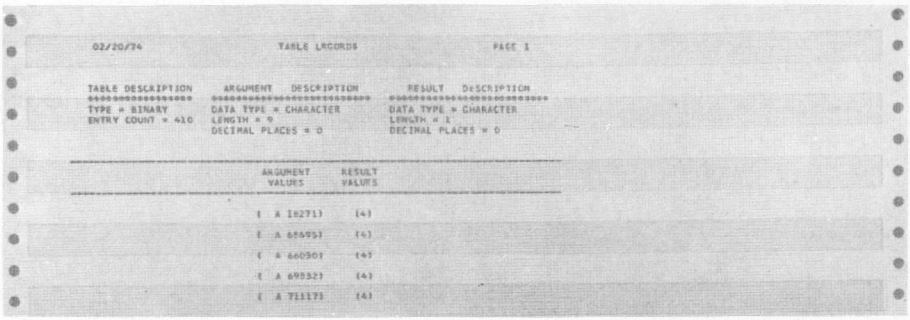

02/20/74 TABLE LKCORDS PAGE 1

TABLE DESCRIPTION
TYPE = BINARY
ENTRY COUNT = 410

ARGUMENT DESCRIPTION
DATA TYPE = CHARACTER
LENGTH = 9
DECIMAL PLACES = 0

RESULT DESCRIPTION
DATA TYPE = CHARACTER
LENGTH = 1
DECIMAL PLACES = 0

ARGUMENT VALUES	RESULT VALUES
(A 102711	(4)
(A 656951	(4)
(A 660301	(4)
(A 698321	(4)
(A 711171	(4)

Fig. 1-11.–Típica impresión de caracteres codificados.

de introducir los datos directamente al computador. Los caracteres almacenados en tarjetas o cintas perforadas entran posteriormente al computador a través de una lectora.

Hay que resaltar que los códigos empleados pueden ser arbitrarios y no estándar. Un código, diferente a los comentados, puede ser diseñado por cualquier usuario, según su conveniencia o eficacia en la aplicación a un problema particular. Sólo, cuando tiene que ser usado en diferentes circuitos y tratado por muchos equipos, llega a ser necesario un código estándar.

Se ofrece a continuación un listado completo de los códigos ASCII y EBCDIC en las figuras 1-12 y 1-13. El código ASCII emplea 7 bits de información y uno de paridad; el EBCDIC es una variante del BCD y dispone de 8 bits.

PALABRAS Y BYTES

Con el fin de procesar datos en un sistema digital, así como de transmitirlos entre dos sistemas, es común agrupar los números, caracteres del alfabeto, símbolos y otros tipos de datos, como grupos de bits, en un formato establecido, es decir, de longitud uniforme. Estos grupos se llaman *palabras o bytes.* La longitud o número de bits, que componen un byte, puede variar de un sistema a otro. En la figura 1-14 se muestra un formato, en el que la palabra consta de 16 bits y el byte de 8, de forma que una palabra contiene a 2 bytes.

Dentro de cada byte, los bits están ordenados desde el *bit de menos peso o significado* en un extremo, al *bit de más peso o significado* en el otro. Esto quiere decir que, si el byte representa a un número, en la figura 1-14 el dígito de la derecha tiene un peso o valor 1 en decimal, el siguiente dígito tiene un peso de 2, el siguiente de 4, luego 8, 16, ..., subiendo el valor de esta forma hasta que el octavo dígito alcance el de 128. También se usa la expresión de "bit de más o menos peso" en bytes y palabras, que representan caracteres del alfabeto, aunque en estos casos los dígitos carezcan de valor. De esta manera se define el orden con que se transmiten los bits, dentro de la palabra o byte, a través de una simple línea (en serie), o a través de 8 ó 16 (en paralelo).

Como consecuencia de la variación entre los computadores en su constitución y estructura, a menudo la longitud de la palabra difiere de unos a otros. En general, una palabra consta de uno o más bytes, y éstos tienen una longitud de 8 bits, pudiendo ser la longitud de la palabra de 8, 16, 24, 32, 56 ó 64 bits. En ciertos computadores se denomina *página* a un grupo de palabras, por ejemplo 256.

COLUMNA	0	1	2	3	4	5	6	7
FILA / BITS 4321 765	000	001	010	011	100	101	110	111
0	0000	NUL	DLE	SP	0	@	P	\	p
1	0001	SOH	DC1	!	1	A	Q	a	q
2	0010	STX	DC2	"	2	B	R	b	r
3	0011	ETX	DC3	#	3	C	S	c	s
4	0100	EOT	DC4	$	4	D	T	d	t
5	0101	ENQ	NAK	%	5	E	U	e	u
6	0110	ACK	SYN	&	6	F	V	f	v
7	0111	BEL	ETB	'	7	G	W	g	w
8	1000	BS	CAN	(	8	H	X	h	x
9	1001	HT	EM	)	9	I	Y	i	y
10	1010	LF	SUB	*	:	J	Z	j	z
11	1011	VT	ESC	+	;	K	[	k	{
12	1100	FF	FS	,	<	L	\	l	¦
13	1101	CR	GS	–	=	M	]	m	}
14	1110	SO	RS	.	>	N	^	n	~
15	1111	SI	US	/	?	O	—	o	DEL

Ejemplo: Código para A = 7 - - - - - 1 = 100 0001

Significado en inglés de las leyendas de las columnas 0 y 1

NUL	*Null*
SOH	*Start of Heading*
STX	*Start of Text*
ETX	*End of Text*
EOT	*End of Transmission*
ENQ	*Enquiry*
ACK	*Acknowledge*
BEL	*Bell (audible signal)*
BS	*Backspace*
HT	*Horizontal Tabulation (punched card skip)*
LF	*Line Feed*
VT	*Vertical Tabulation*
FF	*Form Feed*
CR	*Carriage Return*
SO	*Shift Out*
SI	*Shift In*
DLE	*Data Link Escape*
DC1	*Device Control 1*
DC2	*Device Control 2*
DC3	*Device Control 3*
DC4	*Device Control 4*
NAK	*Negative Acknowledge*
SYN	*Synchronous Idle*
ETB	*End of Transmission Block*
CAN	*Cancel*
EM	*End of Medium*
SUB	*Substitute*
ESC	*Escape*
FS	*File Separator*
GS	*Group Separator*
RS	*Record Separator*
US	*Unit Separator*
DEL	*Delete*

Fig. 1.12.–Listado completo del código ASCII.

FILA (HEX)	BITS 4567 / 0123	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	COLUMNA (HEX)	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
0	0000	NUL	DLE	DS		SP	&	-						{	}	\	0
1	0001	SOH	DC1	SOS						a	j	~		A	J		1
2	0010	STX	DC2	FS	SYN					b	k	s		B	K	S	2
3	0011	ETX	DC3							c	l	t		C	L	T	3
4	0100	PF	RES	BYP	PN					d	m	u		D	M	U	4
5	0101	HT	NL	LF	RS					e	n	v		E	N	V	5
6	0110	LC	BS	EOB ETB	UC					f	o	w		F	O	W	6
7	0111	DEL	IL	PRE ESC	EOT					g	p	x		G	P	X	7
8	1000		CAN							h	q	y		H	Q	Y	8
9	1001	RLF	EM						\	i	r	z		I	R	Z	9
A	1010	SMM	CC	SM		¢	!	¦	:								
B	1011	VT				.	$	'	#								
C	1100	FF	IFS		DC4	<	*	%	@								
D	1101	CR	IGS	ENQ	NAK	(	)	—	'								
E	1110	SO	IRS	ACK		+	;	>	=								
F	1111	SI	IUS	BEL	SUB	|	¬	?	"								

Ejemplo: Código para letra D = bits 0 ..... 7 = 1100 0100

Significado en inglés de las columnas 0 y 4

NUL	*Null*	*CC*	*Cursor Control*
SOH	*Start of Heading*	*IFS*	*Interchange File Separator*
STX	*Start of Text*	*IGS*	*Interchange Group Separator*
ETX	*End of Text*	*IRS*	*Interchange Record Separator*
PF	*Punch Off*	*IUS*	*Interchange Unit Separator*
HT	*Horizontal Tab*	*DS*	*Digit Select*
LC	*Lower Case*	*SOS*	*Start of Significance*
DEL	*Delete*	*FS*	*Field Separator*
RLF	*Reverse Line Feed*	*BYP*	*Bypass*
SMM	*Start of Manual Message*	*LF*	*Line Feed*
VT	*Vertical Tabulation*	*EOB/ETB*	*End of Block/End of Transmission Block*
FF	*Form Feed*	*PRE/ESC*	*Prefix/Escape.*
CR	*Carriage Return*	*SM*	*Set Mode*
SO	*Shift Out*	*ENQ*	*Enquiry*
SI	*Shift In*	*ACK*	*Acknowledge*
DLE	*Data Link Escape*	*BEL*	*Bell*
DC1	*Device Control 1*	*SYN*	*Synchronous Idle*
DC2	*Device Control 2*	*PN*	*Punch On*
DC3	*Device Control 3*	*RS*	*Reader Stop*
RES	*Restore*	*UC*	*Upper Case*
NL	*New Line*	*EOT*	*End of Transmission*
BS	*Backspace*	*DC4*	*Device Control 4*
IL	*Idle*	*NAK*	*Negative Acknowledge*
CAN	*Cancel*	*SUB*	*Substitute*
EM	*End of Medium*	*SP*	*Space*

Fig. 1-13.–Listado completo del código EBCDIC.

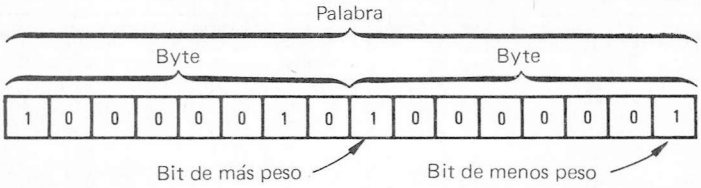


Fig. 1-14.–Formato de una palabra de 16 bits, que consta de 2 bytes de 8 bits cada uno.

El hecho de que las longitudes de los bytes y palabras sea de 8, 16, etc. no es arbitrario y tiene una razón de ser. En principio, se requieren 4 bits para escribir cualquier número del 0 al 9 en forma binaria codificada. Como el sistema decimal se emplea frecuentemente, es práctico que los bytes y palabras sean múltiplos de 4. En segundo lugar, se necesitan de 120 a 130 combinaciones diferentes para definir todos los caracteres del alfabeto, los 10 números y el grupo que comprende símbolos y comandos. Como con 7 bits se pueden definir 128 combinaciones diferentes y, a menudo, un bit se emplea para la prueba de paridad, el byte de 8 bits resulta el de longitud práctica más útil.

INSTRUCCIONES, DIRECCIONES Y OTROS DATOS

Además de números, letras y símbolos, las palabras de datos pueden contener también otros tipos de información. En general, para todo sistema se pueden reconocer tres clases diferentes de información:

Instrucción: Dice lo que se debe hacer con los datos, que siguen detrás de ella. Una instrucción puede representar una suma, un almacenamiento de datos, la impresión del teletipo, etc.

Dirección: Determina dónde se deben localizar los datos en la memoria.

Datos y caracteres alfanuméricos: A los que se refieren y manejan las instrucciones y direcciones.

Cualquier palabra puede contener una clase de información o una combinación de varias. Por ejemplo, el byte de más peso de la palabra puede contener una instrucción o una dirección y los bytes que quedan, pueden ser los datos. Cuando una palabra queda dividida de este modo, es corriente referirse al *campo de instrucción, campo de dirección* y *campo de datos.*

Cuando un sistema es muy simple, o se ha construido para un solo objetivo, de forma que siempre realiza la misma operación, almacenando una y otra vez los resultados en el mismo sitio, es posible que no necesite

ni instrucciones ni direcciones. En el diseño de un sistema se pueden presentar una de estas dos alternativas: construir los circuitos más versátiles y complejos, para permitir que las palabras de instrucciones y direccionamiento dirijan por el sistema el flujo de datos, o bien, construir un equipo muy especializado y no usar palabras de instrucciones ni de direcciones. La elección clásica, que se presenta al diseñar circuitos de computadores, estriba entre hacer más complejo y caro el HARDWARE, o sea, la configuración física del circuito, o, en su lugar, usar más SOFTWARE, es decir, mayor potencia para las instrucciones, que rigen el comportamiento del circuito.

INTERCAMBIO DE DATOS

Cuando se transfiere un dato entre dos dispositivos cualesqueira, tal como entre un teletipo y una computadora, o entre una computadora y una memoria de discos, hay que decodificarlos mediante el dispositivo de entrada adecuado. Una gran computadora, que trabaja con palabras de 64 bits, no es directamente compatible con un periférico, que transmite o recibe palabras de sólo 8 bits. De igual manera, un periférico, que saca sus datos en serie (un bit en cada momento), tampoco es compatible con una computadora, que sólo acepta los datos en paralelo (muchos bits a la vez). Los datos en serie deben convertirse a datos en paralelo antes de entrar en la computadora. Este tipo de conversión de datos se lleva a cabo en un circuito de adaptación, llamado técnicamente de "interface", tal como queda representado en la figura 1-15.

Fig. 1-15.–Utilización de un elemento de interface.

Hay otro tipo de conversión, que ocurre cuando, por ejemplo, la computadora saca los datos más deprisa de lo que puede aceptar el periférico. También en este caso hace falta un circuito adaptador, que almacene los datos del computador en una memoria intermedia (buffer) hasta que los absorba el periférico. En resumen, para cambiar la longitud de las palabras y la cantidad de datos se usan circuitos de interface, que también se emplean para producir otros cambios de información entre la computadora y los periféricos, como pueden ser, la demanda de información, señales de ocupación e interrupción, etc.

CONTROL DE PARIDAD Y CORRECCION DE ERROR

Un problema usual en el almacenamiento de datos y en las redes de comunicación es el de reconocer un error de los datos. Por ejemplo, en un sistema, que usa cinta magnetofónica para almacenar una secuencia de palabras binarias, es importante tener un indicador, que nos diga cuándo se transfiere una palabra errónea a/o desde la lectora-grabadora. La lectura incorrecta de un bit en una determinada palabra, un 0 en lugar de un 1 ó viceversa, produce un importante error, que conviene detectar.

El método más común para detectar un error en una palabra es: primero, contar el número de bits lógicos 1, que hay en un carácter de 7 bits. Si el número de 1 existentes es impar (1, 3, 5 ó 7), se añade al grupo un nivel lógico 1 y así la palabra completa de 8 bits tiene un número par de bits 1. Si, por el contrario, el número de bits 1 es par, se añade un bit 0 en la octava posición, con lo que la palabra de 8 bits tiene un número par de bits 1 en ella. El objetivo es conseguir siempre que la palabra de 8 bits contenga un número par de bits 1. Cuando estas palabras de 8 bits se transfieren de un dispositivo a otro, a la entrada de este último se deben contar los bits 1 que llegan en cada palabra y verificar si sigue habiendo un número par de ellos. Si esto no es así, se genera una señal de "error de paridad". A este proceso se denomina "Control de Paridad" y al tipo de paridad descrita se le denomina "paridad par". Si se desea comprobar la paridad impar, cada palabra binaria contendrá un número impar de bits 1. Ambos tipos de paridad se emplean frecuentemente.

Téngase en cuenta que, en el caso de que se pierdan dos bits lógicos 1 en la misma palabra de 8 bits, la palabra seguirá manteniendo la misma paridad y este control quedará invalidado. En principio, este supuesto es cierto, pero los dispositivos de memoria se caracterizan por su gran exactitud, lo que hace muy improbable que presenten error dos bits de la misma palabra.

Añadiendo más bits a la palabra y confeccionando circuitos más complejos de codificación y detección de errores, se puede perfeccionar el control de la paridad. Se puede llegar a conocer cuál de los bits es el erróneo e, incluso, hay circuitos capaces de corregir el error y hacer que el flujo de palabras suceda ininterrumpidamente (en las grandes computadoras).

CAPITULO 2

Teoría general sobre la arquitectura de los ordenadores digitales

INTRODUCCION

Los ordenadores digitales son aquéllos que están fabricados esencialmente a base de circuitos electrónicos digitales. Puede afirmarse que la aplicación más vasta, así como la de mayor consumo de los elementos electrónicos digitales, es la de los ordenadores. También existen ordenadores analógicos, apropiados, en su tiempo, para resolver problemas específicos y que están basados en un circuito de control en lazo cerrado (realimentación), a base de Amplificadores Operacionales, que proporcionan una salida, que se compara continuamente con las entradas y se corrige automáticamente. Finalmente, los ordenadores híbridos disponen de circuitos analógicos y digitales.

La presencia de los ordenadores digitales en el mercado mundial es masiva y, por esta razón, se pueden considerar los más representativos en su área. Por otra parte, si se conocen todos los bloques que conforman la Electrónica Digital, es relativamente sencillo comprender la arquitectura y funcionamiento del ordenador digital, puesto que está construido esencialmente por una gran parte de los mencionados bloques.

Un computador digital es una máquina capaz de procesar y manipular los datos e informaciones, que se le suministran o que tiene almacenados, siguiendo las reglas de la técnica digital. De esta manera, el computador digital recibe informaciones desde su memoria y desde sus elementos periféricos, que le relacionan con el mundo exterior (teletipos, circuitos electrónicos, relés, sensores, etc.) y los procesa de acuerdo con una secuencia de instrucciones, programada previamente y almacenada en la memoria del ordenador. En la fase del proceso de datos, el ordenador tiene capacidad para realizar operaciones aritméticas y lógicas, desplazamientos, transferencias, etc. para, una vez obtenidos los resultados, entregarlos al exterior en el código binario con que trabaja la máquina, quedando por cuenta de los circuitos de adaptación (Entradas-Salidas) y de

los periféricos, traducirlos y ofrecerlos al ser humano de una forma más asequible (impresoras, pantallas, display o presentadores luminosos, etc.).

Se denomina programa a la lista de instrucciones que debe desarrollar el ordenador, generalmente de forma secuencial, para resolver una aplicación específica de tratamiento de datos. Por este motivo, cada ordenador dispone de un juego de instrucciones, que es capaz de interpretar y ejecutar.

Normalmente estas instrucciones tienen relación y se han diseñado de acuerdo con las aplicaciones generales a las que se destina el ordenador.

ARQUITECTURA FUNDAMENTAL DEL ORDENADOR DIGITAL

Todo ordenador digital consta de 4 bloques funamentales: Unidad de Control, Unidad Operativa (ALU), Memoria y Unidad de Entradas y Sali-

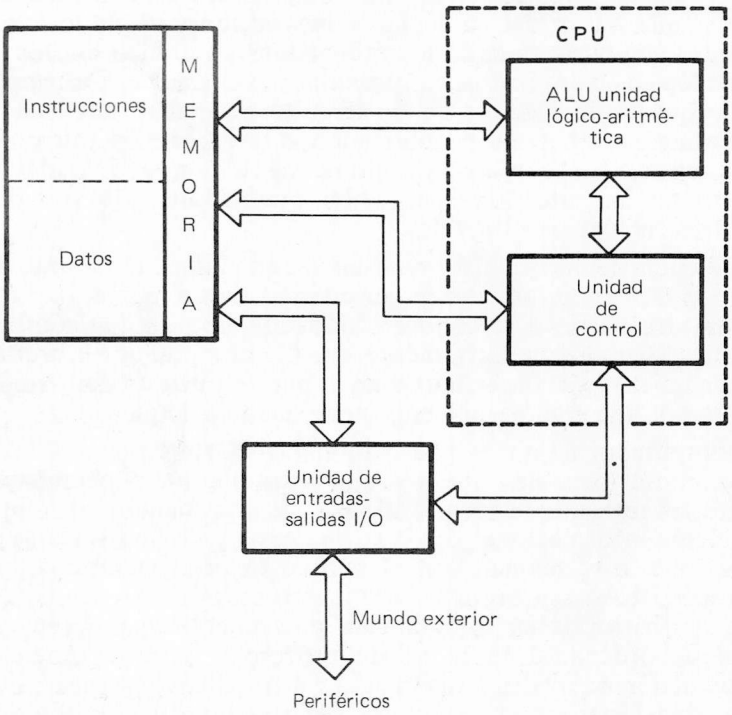


Fig. 2-1.–Arquitectura fundamental del ordenador digital. Hay ocasiones que la unidad de I/O accede a Memoria a través de la CPU.

das. Además, dispone de una serie de registros de trabajo para almacenar datos, direcciones u otras informaciones y un Generador de impulsos de reloj, denominado técnicamente Clock, encargado de la sincronización del trabajo de la máquina. La arquitectura fundamental del ordenador por bloques, se presenta en la figura 2-1.

Al conjunto formado por la Unidad de Control, la ALU, los registros de trabajo y el generador de impulsos de reloj, se le denomina generalmente Unidad Central de proceso (UCP) derivado de la expresión inglesa Control Process Unit (CPU). Cuando todo este conjunto se fabrica sobre un chip de un circuito integrado, recibe el nombre de "microprocesador".

La UCP crea y controla el flujo de datos, que circula por el ordenador a partir de las instrucciones recibidas de la memoria, que sirven para indicar las operaciones o tratamiento a realizar sobre los datos recibidos desde el exterior o previamente almacenados en la memoria. Por lo tanto, se puede considerar que la memoria junto con la UCP forman la parte más importante del ordenador, quedando sólo para completar su estructura, la Unidad de Entradas y Salidas, cuya misión es la de adaptar los códigos de las informaciones que se intercambian el procesador y los periféricos, los cuales relacionan al sistema con el mundo exterior.

A continuación se profundiza un poco más en cada uno de los 4 bloques fundamentales del ordenador digital.

MEMORIA

El programa o secuencia de instrucciones, que debe seguir la máquina para realizar el procesamiento de los datos, está almacenado en una parte de la memoria, denominada "memoria de instrucciones" para diferenciarla del resto de la misma, que se emplea para guardar datos y resultados de forma temporal.

Siguiendo la línea de fabricación moderna de microordenadores, se supone siempre que la memoria está construida a base de circuitos integrados. La figura 2-2 muestra los bloques o partes características de toda memoria.

Por lo general, la memoria de los ordenadores se compone de dos partes: 1.ª) la denominada "central" o "principal", comprendida junto con la UCP en el procesador, en la que suelen estar cargados el programa Supervisor y los de trabajo y es la que interviene en el procesamiento de todas las informaciones; 2.ª) la memoria "periférica", de capacidad ampliable (unidades de discos por ejemplo), donde se guarda gran cantidad de programas y de datos.

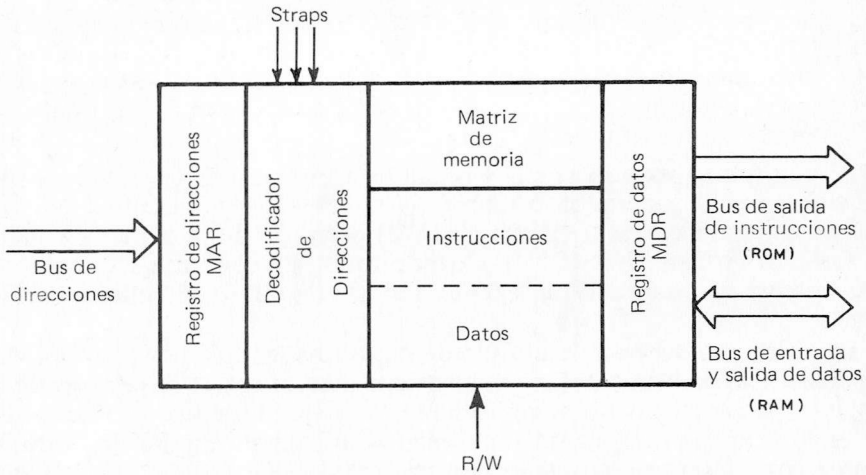


Fig. 2-2.–Partes que componen la memoria del ordenador. Normalmente el bus de salida de instrucciones y el de entrada-salida de datos son el mismo.

La información, que recibe la memoria por el "bus de direcciones", es un conjunto de bits lógicos, tantos como líneas tenga el bus, que seleccionan la posición de la memoria a la que se accede. El decodificador de direcciones se encarga de elegir una posición de la matriz de la memoria, decodificando la información, que ha llegado por el bus de direcciones y se ha almacenado temporalmente en el Registro de Direcciones, MAR. Como generalmente la memoria consta de bastantes chips, existen unas líneas de activación, llamadas "straps" que seleccionan el chip adecuado, otras veces sólo existe una línea de selección de chip (CS). Una vez seleccionada la posición de memoria, si ésta constituye una ROM de sólo lectura, el contenido de la mencionada posición se almacena temporalmente en el Registro de Datos, que, en este caso, son intrucciones y sale finalmente por el bus de salida de instrucciones. Si la memoria es de tipo RAM, de lectura y escritura, se precisa un bus bidireccional, que permita, bien la entrada de datos al MDR y su posterior grabación en la posición seleccionada, o bien la salida de la información contenida en dicha posición. La selección entre la lectura o escritura de una posición de memoria se efectúa por la línea R/W (Read/Write).

UNIDAD DE CONTROL

La Unidad de Control, como puede apreciarse en la figura 2-3, recibe secuencialmente las instrucciones desde la memoria por el bus de instruc-

ciones, almacenándolas en el Registro de Instrucciones (IR). Desde IR las instrucciones pasan al Decodificador de Instrucciones, el cual se encarga de interpretarlas o decodificarlas y producir una serie de impulsos de gobierno y control en el bloque Generador de impulsos o Secuenciador. Estos impulsos regulan a los elementos de la máquina, que participan en la ejecución de la instrucción.

La Unidad de Control, además de decodificar las instrucciones y de generar los impulsos de control, incrementa sincrónicamente a un contador, llamado de programa (PC) cada vez que se ejecuta una instrucción, con objeto de que se quede señalando la siguiente instrucción. En la figura 2-3 se aprecia la configuración de la Unidad de Control.

En la figura 2-3 puede apreciarse que, cuando la instrucción hace referencia a una posición de memoria, puede obtenerse desde la Unidad de Control una dirección de la memoria de datos. También la decodificación de una instrucción puede llevar aparejado un sustancial cambio en el valor del Contador de Programa (PC), que, en casos normales, se limita a incrementarse una unidad por cada instrucción. Este tipo de instrucciones serán las que posibiliten la rotura de la secuencia ordenada de las instrucciones de un programa.

Normalmente, una instrucción consta de varios pasos elementales, denominados "microinstrucciones", cuyos códigos suelen estar almacenados en una memoria ROM, de sólo lectura, de tal forma que la codificación de una determinada instrucción produce la ejecución de las diferentes microinstrucciones de que consta y que se han microprogramado previamente.

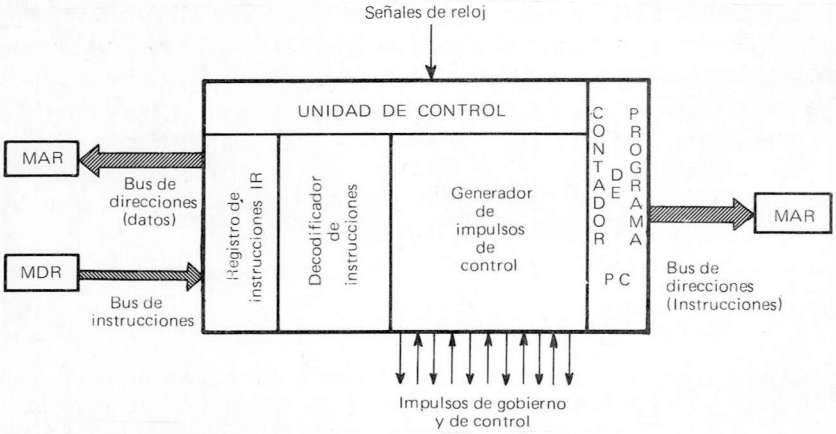


Fig. 2-3.–Estructura fundamental de la Unidad de Control.

UNIDAD LOGICO-ARITMETICA. ALU

La Unidad Operativa o Unidad Lógico-Aritmética (ALU) es la encargada del procesamiento lógico y aritmético de los datos, según el carácter que determine cada instrucción. En la figura 2-4 se muestra la estructura de la Unidad Operativa, que, como puede apreciarse, consta de dos bloques fundamentales, al menos al realizar una consideración de tipo general. Dichos bloques son el Registro ACUMULADOR (A) y la Unidad Lógico-Aritmética (ALU).

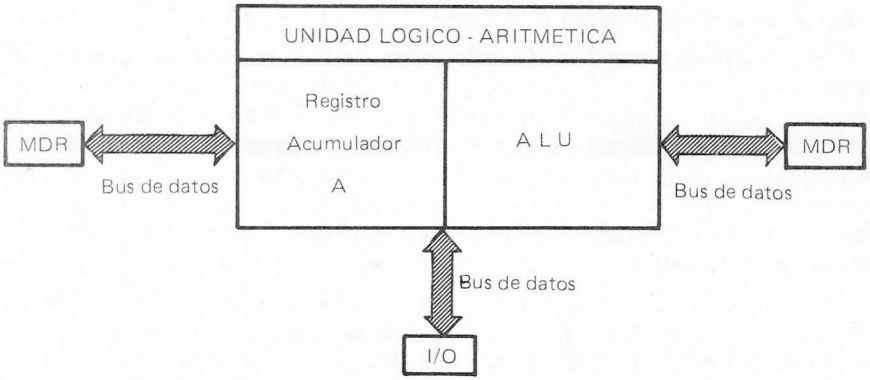


Fig. 2-4.–Estructura básica de la Unidad Operativa.

La capacidad y potencia de la Unidad Operativa en el tratamiento de los datos depende exclusivamente de las características de la ALU que la conforma.

Uno de los datos que maneja la ALU, llega siempre a través del Registro Acumulador, ligado a ella y de mucha flexibilidad. Por lo general, el otro dato llega a la ALU por el bus de datos, bien desde la memoria, o desde los módulos de entrada y salida. El resultado de la operación ejecutada por la ALU se carga en el Registro Acumulador.

UNIDAD DE ENTRADAS Y SALIDAS

Es la encargada de suministrar al ordenador desde el exterior los datos a procesar, así como de sacar al exterior los resultados obtenidos en la forma adecuada. Por ejemplo, si el procesador está conectado a un teletipo (TTY) como periférico, la unidad de entradas y salidas es la encargada

de adaptar entre sí a los dos elementos. Si el TTY trabaja en el código ASCII, al teclear un programa sobre él, cada tecla genera unos impulsos en dicho código, que, previa decodificación, son impresos sobre papel o almacenados sobre cinta perforada. Los impulsos generados por el TTY no son aceptados por el procesador, debido a sus niveles eléctricos y a su código (ASCII). Se precisa un circuito capaz de traducir los impulsos del TTY al lenguaje, que sea capaz de entender la máquina. Similarmente, una vez procesados los datos, los resultados que ofrece el ordenador están codificados en binario y no son inteligibles para el TTY, siendo misión de la unidad de entradas y salidas la correcta adaptación.

DIAGRAMA POR BLOQUES DE UN ORDENADOR DIGITAL

Un diagrama simplificado de la arquitectura de un ordenador digital, se muestra en la figura 2-5, en la que se aprecia la interconexión de sus 4 bloques fundamentales, analizados hasta el momento.

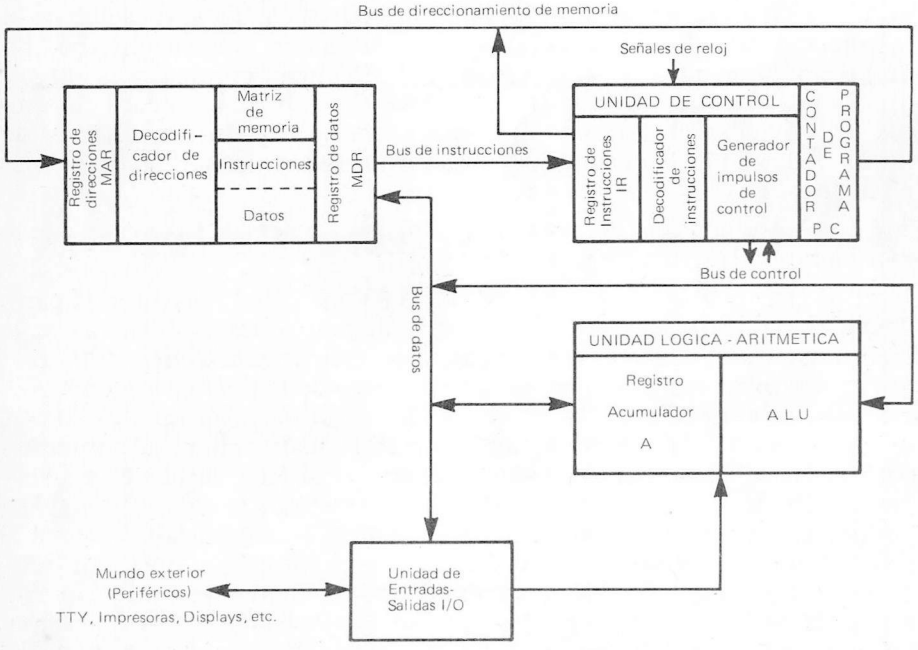


Fig. 2-5.—Diagrama por bloques, simplificado, de un ordenador digital.

Como se aprecia en el diagrama por bloques, los intercambios de informaciones entre los diversos bloques, así como con el exterior, quedan canalizados a través de 3 buses (conjuntos de líneas): el de direccionamiento de la memoria, el de datos, por el que normalmente también circulan las instrucciones, (aunque en la figura 2-5 se han dibujado independientes), y finalmente por el bus de control.

El diagrama por bloques del ordenador permite un análisis elemental de la ejecución de una instrucción de un programa, por ejemplo, la suma de dos datos. Inicialmente, el Contador de Programa (PC) deberá ser cargado con el valor de la dirección de la memoria, en donde está almacenada la instrucción a ejecutar. El contenido del PC pasa a través del bus de direccionamiento y se selecciona una posición de memoria, en la cual está grabado el código de la instrucción de suma. Dicha instrucción pasa a la Unidad de Control por el bus de instrucciones y se almacena en el IR, después se decodifica y, a continuación, se genera una serie de impulsos de control, que regulan la ejecución de la operación en la ALU. Uno de los datos a sumar estará contenido en el Acumulador, mientras que el otro llegará a la ALU desde la memoria, previo direccionamiento de la misma. Este direccionamiento ha sido obtenido de la decodificación de la instrucción, que llevaba la posición de memoria donde se encontraba uno de los dos sumandos. Finalmente, la ALU efectuará la suma, y el resultado se almacenará en el Registro Acumulador, en donde quedará disponible, bien para sacarlo al exterior con una nueva instrucción, a través de las unidades de entrada y salida, o bien para almacenarlo temporalmente en la memoria.

Como resumen, el funcionamiento elemental del ordenador se puede considerar de la siguiente forma:

El programa o secuencia de instrucciones, está almacenado en la parte de memoria destinada a este fin, ocupando posiciones correlativas de la misma. Direccionada la primera instrucción mediante el Contador de Programa, en principio, hay una fase de *búsqueda* de la instrucción en la memoria y, posteriormente, otra fase de *ejecución* de la misma. Localizada en la memoria la posición donde se halla la instrucción, su contenido pasa al Registro de Instrucciones a través del MDR y el bus de instrucciones. Desde este registro se introduce el código de la instrucción al Decodificador de Instrucciones, el cual interpreta su significado y, según el mismo, origina una serie de impulsos de control desde el Generador o Secuenciador, que gobierna la actuación de los diferentes elementos, que intervienen en la ejecución de la instrucción. Puede haber instrucciones que hagan referencia a datos guardados en la memoria, en cuyo caso, se obtiene la dirección de la misma desde el propio IR. Esto explica la conexión entre IR y el bus de direccionamiento.

Cada vez que se inicia la realización de una instrucción, que, como se indicó, puede estar compuesta por varias microinstrucciones, el Contador de Programa queda incrementado y pasa a direccionar la siguiente instrucción del programa.

Una de las operaciones más corriente del ordenador consiste en realizar una operación lógica o aritmética con los datos. En este caso, su implementación se lleva a cabo por la ALU. Uno de los datos se almacena en el registro Acumulador y el otro llega directamente a la otra entrada de la ALU, ambos datos pueden ser enviados bien desde la memoria, o bien desde los elementos de entrada y salida. Finalmente, el resultado de la operación queda disponible en el Acumulador para, posteriormente, almacenarlo en la memoria o para sacarlo al exterior.

FASES DE UNA INSTRUCCION

Cada instrucción, que realiza el ordenador, se lleva a cabo en dos fases: "fase de búsqueda" y "fase de ejecución".

Fase de búsqueda: En esta fase el Contador de Programa (PC) contiene la dirección de la posición de la memoria en donde se encuentra el código de la instrucción. Dicho contenido se transfiere por el bus de direccionamiento al Registro MAR y, una vez decodificado y seleccionado el contenido de la posición elegida, se carga al registro MDR. Desde el MDR se transfiere por el bus de instrucciones, de tantas líneas como bits tenga la palabra, al registro de instrucciones IR de la Unidad de Control. La figura 2-6 muestra gráficamente los pasos de que consta la fase de búsqueda de una instrucción.

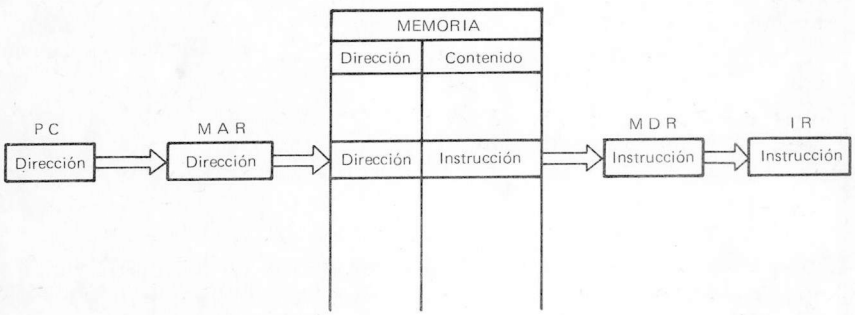


Fig. 2-6.–Movimiento de los contenidos entre las unidades que participan en la fase de búsqueda de una instrucción.

Fase de ejecución:. Acabada la fase de búsqueda, el contenido del IR, es decir, el código de la instrucción, es interpretado en el Decodificador de Instrucciones, al mismo tiempo que se incrementa el Contador de Programa. La decodificación de la instrucción produce en la Unidad de Control la generación de los impulsos necesarios para llevar a cabo la instrucción. El PC, incrementado, queda preparado para comenzar la fase de búsqueda de la siguiente instrucción.

REALIZACION DE UN PROGRAMA EN EL ORDENADOR

Analizada la arquitectura de un ordenador digital, para acercarse mejor a la comprensión de su funcionamiento, se expone a continuación la forma con que opera en la realización de un simple programa, que pueda ser entendido incluso por quienes no conocen nada sobre programación.

Para realizar un programa de un ordenador, primero y una vez conocido a fondo el problema, que se trata de resolver, se confecciona un diagama gráfico que represente todas las operaciones elementaleś que hay que realizar hasta conseguir el fin que se pretende. A este dibujo se le llama "diagrama de flujo" u "ordinograma" y, como introducción al mismo, sólo se presentan sus dos símbolos fundamentales, uno que expresa la realización de una operación y otro la toma de una decisión, dibujándose ambos en la figura 2-7.

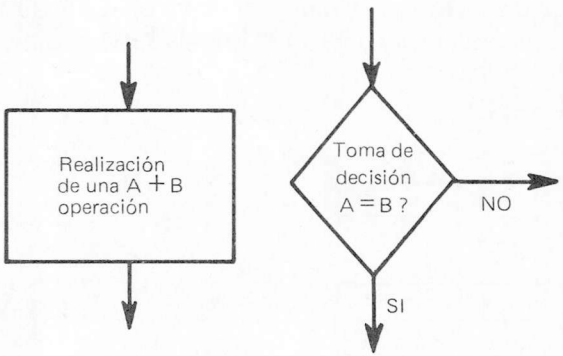


Fig. 2-7.–Símbolos fundamentales del "diagrama de flujo".

El ordinograma debe desmenuzar el programa en los pasos elementales, que sea posible realizar, mediante las instrucciones que dispone el ordenador. Es decir, al confeccionar un ordinograma hay que tener muy en cuenta las instrucciones que es capaz de interpretar el Decodificador de Instrucciones de la Unidad de Control.

Ejemplo

Se desea conocer el número total de piezas, que han producido dos máquinas, llamadas A y B. Cada máquina dispone de un contador que junto con el apropiado circuito de adaptación, introduce constantemente el número de piezas fabricadas hasta el momento elegido. Los valores de los contadores se almacenan en determinadas posiciones de la memoria. El ordenador ha de tomar los datos ofrecidos por los dos contadores, sumarlos y visualizar el resultado en un presentador numérico. Un ordinograma básico para este ejemplo, se presenta en la figura 2-8.

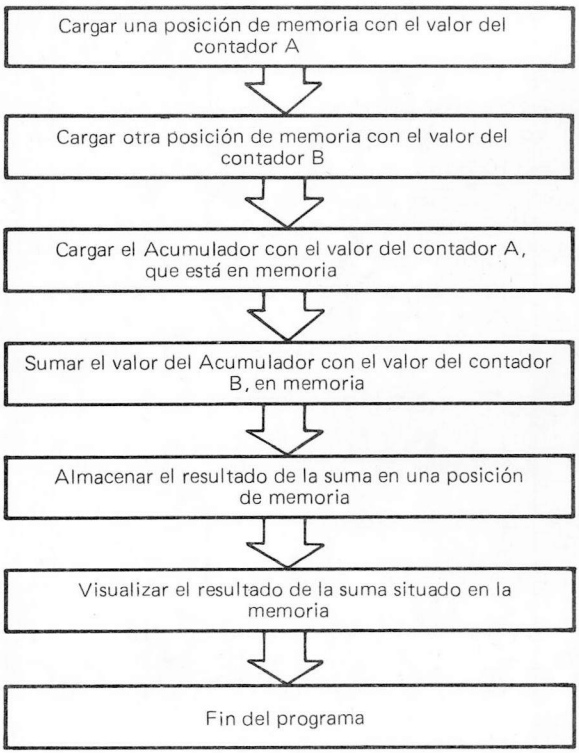


Fig. 2-8.–Ordinograma elemental para resolver el ejemplo de las máquinas.

Al objeto de comprobar la misión y el significado de las operaciones expresadas, mediante el símbolo de toma de decisión, supongamos que, respecto al ejemplo primitivo, se pretenda que la visualización del resultado sólo debe efectuarse, cuando la suma de los valores A + B = 100; caso de ser menor, se seguirá efectuando el testado hasta que se alcance esta cifra, como queda reflejado en el nuevo ordinograma de la figura 2-9.

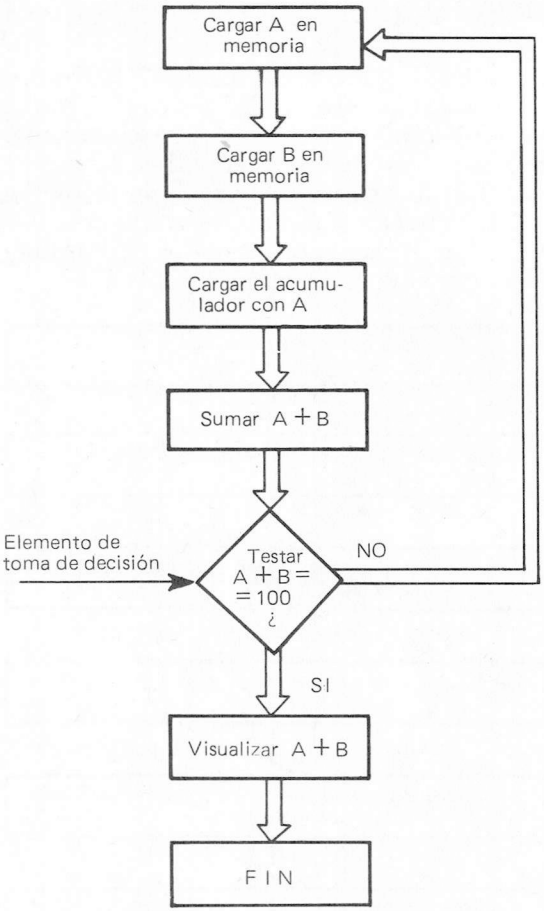


Fig. 2-9.—Ordinograma con una instrucción de toma de decisión.

La operación de toma de decisión representa, en el ejemplo de la figura 2-9, que, si $A + B \neq 100$, se ha de volver al principio del programa rompiendo la secuencia ordenada del mismo. Esta posibilidad es muy importante.

Refiriéndonos al ejemplo inicial, y no al que contiene una toma de decisión, si ya está establecido el ordinograma correspondiente (fig. 2-8), se elige una posición libre de la memoria para contener la instrucción de inicio del programa. Aquí, de forma arbitraria, se ha elegido la 21. Las sucesivas instrucciones del programa quedarán almacenadas en las posiciones correlativas de la memoria, tal como queda reflejado en la figura 2-10.

A continuación, se analiza la forma con que el ordenador actúa y resuelve cada una de las 6 instrucciones de que consta el programa almacenado, según la figura 2-10.

	MAPA DE LA MEMORIA	
	Dirección	**Contenido : Instrucciones o datos**
1ª Instrucción	21	Cargar el Acumulador con el contenido de la posición de memoria 26.
2ª Instrucción	22	Sumar el Acumulador con el contenido de la posición de memoria 27.
3ª Instrucción	23	Almacenar el resultado de la suma en la posición de memoria 28.
4ª Instrucción	24	Visualizar el resultado de la suma
5ª Instrucción	25	Salto a la posición 29 de la memoria
1er Dato	26	Valor del contador de la máquina A
2º Dato	27	Valor del contador de la máquina B
3º Dato	28	Valor de la suma de A + B
6ª Instrucción	29	Fin de procesamiento

Fig. 2-10.–Distribución de las posiciones de la memoria para contener las instrucciones, que implementan el ordinograma de la figura 2-8.

1.ª instrucción: Carga del acumulador. Dirección de memoria 21.

Con esta instrucción se pretende cargar al Registro Acumulador con el contenido de una locación de memoria, la 26, en la cual está grabado el valor del contador de la máquina A.

a) Para iniciar el programa, el Contador de Programa debe cargarse con la dirección en la que se encuentra la 1.ª instrucción, es decir, debe contener el valor 21.

A través del bus de direccionamiento se transfiere el valor 21, que contiene el PC, el Registro MAR de la memoria. Véase el diagrama por bloques del ordenador, que aparece en la figura 2-5.

b) El Decodificador de direcciones de la memoria selecciona la posición 21, que recibe del MAR. La posición 21 contiene en código binario o código máquina la instrucción de CARGA DEL ACUMULADOR DESDE UNA POSICION DE MEMORIA. La instrucción pasa al IR por el bus de instrucciones y a través del Registro MDR de la memoria.

Los pasos *a*) y *b*) de esta instrucción conforman la fase de búsqueda y se representan gráficamente en la figura 2-11.

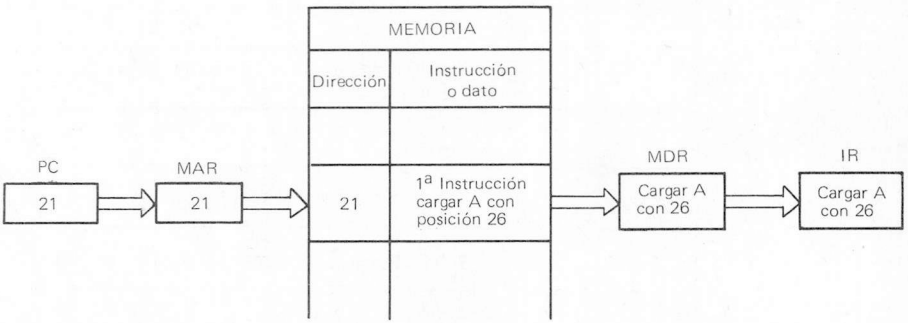


Fig. 2-11.–Pasos que forman la fase de búsqueda de la 1.ª instrucción.

c) A continuación, IR transfiere al MAR la posición 26, como consecuencia de las señales de control, que se han originado tras la decodificación de la instrucción. En dicha posición 26 se encuentra el dato, que ha de cargarse en el Acumulador. Seleccionada y localizada la posición 26 en la memoria, su contenido pasa directamente desde el MDR hasta el

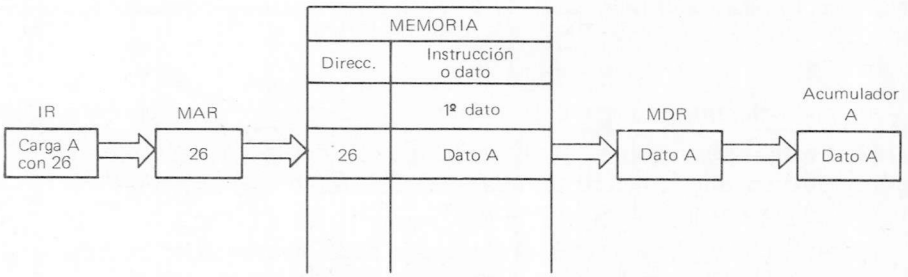


Fig. 2-12.–Expresión gráfica de la fase de ejecución de la 1.ª instrucción.

Acumulador. Véase expresada gráficamente esta fase de ejecución de la 1.ª instrucción del programa en la figura 2-12.

2.ª instrucción: Suma. Dirección de memoria 22.

La fase de búsqueda de esta instrucción se inicia transfiriendo el contenido del PC (22) al MAR, con lo cual se direcciona la posición 22, que contiene el código binario de la instrucción de suma; pasa a continuación al MDR y de allí al IR para completar la fase de búsqueda. Véase la fase de búsqueda expresada gráficamente en la figura 2-13.

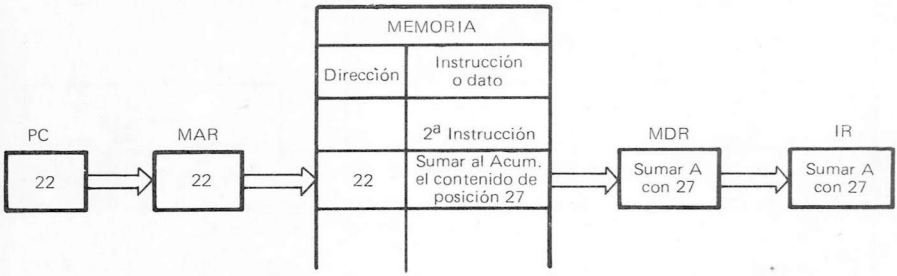


Fig. 2-13.–Pasos de la fase de búsqueda de la 2.ª instrucción de suma.

La fase de ejecución de esta instrucción de suma se inicia, una vez que ha sido decodificada por el Decodificador de instrucciones, colocando desde el IR la dirección 27 en el registro MAR de la memoria. El contenido de la posición 27, que es el valor del contador de la máquina B del ejemplo, pasa al MDR y a continuación se efectúa en la ALU la suma de los contenidos del Acumulador (A) y el valor del MDR (B), depositándose el resultado en el Acumulador, como se expresa simplificadamente en la figura 2-14.

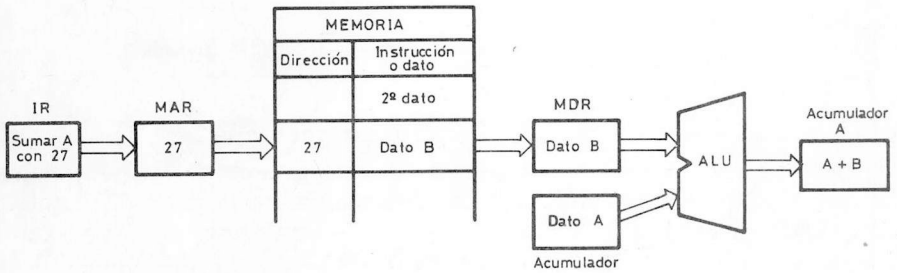


Fig. 2-14.–Desarrollo de la fase de ejecución de la instrucción de suma.

Al comenzar la fase de ejecución de esta instrucción, el PC se ha incremetado una unidad, pasando a contener el valor 23.

3.ª instrucción: Almacenaje. Dirección de memoria 23.

La fase de búsqueda de esta instrucción, por ser totalmente parecida a la fase de búsqueda de las instrucciones precedentes, sólo se expresa gráficamente en la figura 2-15.

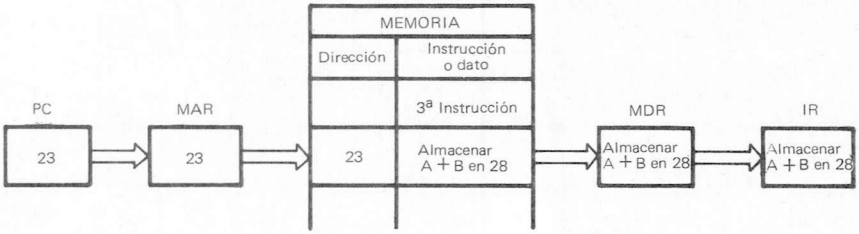


Fig. 2-15.–Fase de búsqueda desarrollada, para la 3.ª instrucción.

En la fase de ejecución, el contenido del Acumulador (A + B) pasa a la posición de memoria número 28, al mismo tiempo que el PC se incrementa a 24. Ver la figura 2-16.

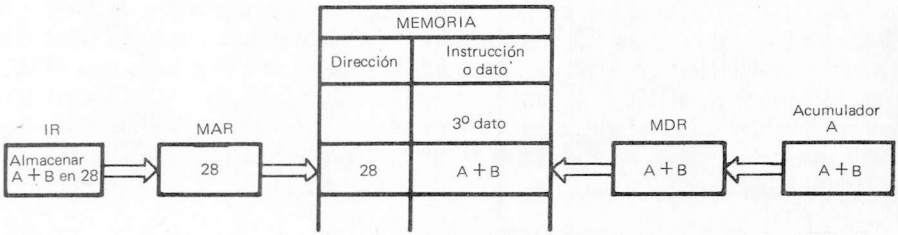


Fig. 2.16.–Desarrollo de la fase de ejecución de la 3.ª instrucción.

4.ª instrucción: Visualización. Dirección de memoria 24.

Con esta instrucción se visualiza el valor de la suma A + B, que se halla depositado en la posición 28 y en el Acumulador. Para ello se comienza sacando el contenido de la posición 28 al exterior, a través de una Unidad de entradas y salidas y después de decodificar adecuadamente su valor, se visualiza en unos presentadores o displays. Tanto la fase de búsqueda, como la de ejecución de esta instrucción se representan gráfica-

mente en la figura 2-17, en la que se ha simplificado la fase de ejecución, visualizando el Acumulador en lugar de la posición 28, puesto que ambas contienen A + B. Es muy frecuente que los contenidos de las posiciones de memoria atraviesen el Acumulador en su camino hacia el exterior.

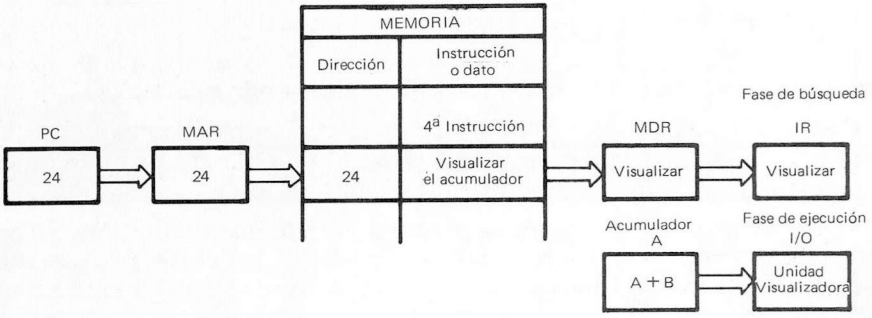


Fig. 2-17.–Fases de búsqueda y de ejecución de la instrucción de visualización.

El PC queda incrementado al valor 25 en la fase de ejecución de la 4.ª instrucción.

5.ª instrucción: Salto. Dirección de memoria 25.

Con esta instrucción de salto se pretende romper la secuencia ordenada en la selección de instrucciones en la memoria. En este caso, en vez de pasar el valor del PC de 25, en que se encuentra en este momento, a su inmediato superior (26), debe pasar a contener el valor 29. Por otra parte, la posición 26 ya está ocupada por un dato temporal. Esta fase de búsqueda se representa en la figura 2-18.

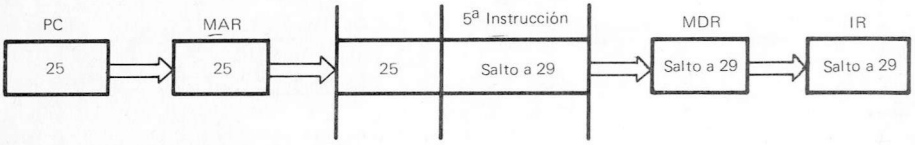


Fig. 2-18.–Fase de búsqueda de la instrucción de salto.

En la fase de ejecución sólo interviene el PC, que se carga con el valor 29, como se indica en la figura 2-19.

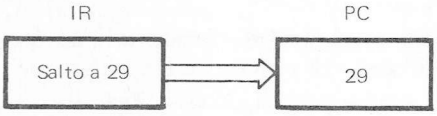


Fig. 2-19.–Fase de ejecución de la instrucción de salto.

6.ª instrucción: Fin o alto. Dirección de memoria 29.

La fase de búsqueda de esta instrucción final, es similar a la de las anteriores instrucciones y se muestra gráficamente en la figura 2-20.

Con esta instrucción se detiene el funcionamiento de la máquina, quedando el Acumulador con el valor A + B, así como la posición 28, mientras el visualizador queda presentando ese mismo valor.

Como se deduce del proceso operativo del ordenador, en todos los intercambios entre registros y diversos elementos, los datos y las instrucciones han de tener el mismo formato o longitud, es decir, el mismo número de bits.

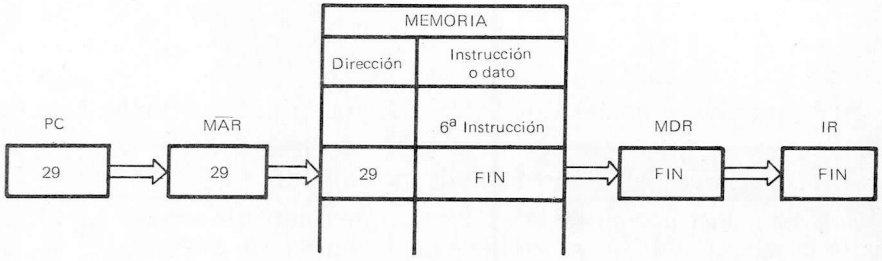


Fig. 2-20.–Fase de búsqueda de la instrucción de Fin o Alto.

La finalidad del tema expuesto ha sido la de ofrecer una visión general sobre la arquitectura del ordenador, así como de su funcionamiento en el procesamiento de las instrucciones y los datos en un determinado programa. Conceptos más amplios y rigurosos, junto con aplicaciones más complejas se discutirán en sucesivos capítulos, en los que se tomará como base al microprocesador, dada la actualidad del mismo, la simplificación que aporta a la arquitectura del sistema computador y las inmensas posibilidades de aplicación en un futuro inmediato.

CAPITULO 3

Las instrucciones en el ordenador

SIGNIFICADO DE LAS INSTRUCCIONES EN EL PROGRAMA ALMACENADO

El trabajo primordial de un ordenador consiste en el procesamiento de los datos, según un programa concreto de instrucciones que tiene almacenado en la memoria.

El procesamiento de los datos se realiza en la UCP, que comprende la Unidad de Control, la Unidad Operativa, basada en la ALU y la memoria principal. El ordenador precisa, además de la UCP, de otra memoria externa, donde se almacenan programas y datos fácilmente transferibles y de las Unidades de entrada y salida, que se encargan de intercambiar información con los periféricos, que relacionan al ordenador con el mundo exterior. Un diagrama elemental de la estructura de un ordenador lo proporciona la figura 3.1.

En el capítulo anterior se estudió la conformación de cada uno de los bloques del ordenador, si bien, por simplificar, sólo se empleó una memoria, que abarcaba a las dos mostradas en la figura 3-1.

Cada ordenador responde exclusivamente al juego de instrucciones, que su Decodificador de Instrucciones, inserto en la Unidad de Control, es capaz de interpretar. Cada fabricante diseña la UCP, según las aplicaciones fundamentales a las que se destine y las características que más desee desarrollar; por este motivo, el juego de instrucciones es diferente en cada tipo de ordenador. Así, por ejemplo, existen en el mercado microprocesadores (UCP integrada en un solo chip) con un juego (set) de instrucciones de alrededor de 50, mientras que otros superan el centenar.

Las palabras, con las que trabajan los ordenadores, disponen de varios bits. Así, es común que los microordenadores utilicen una longitud de palabra de 8 bits, los miniordenadores de 16 y 24 y los ordenadores de 32 y hasta 64 bits. La longitud de la palabra determina ciertas propiedades al ordenador; por ejemplo, si es de 8 bits, las posiciones de la memoria pueden almacenar 8 bits, los buses, por los que circulan instruccio-

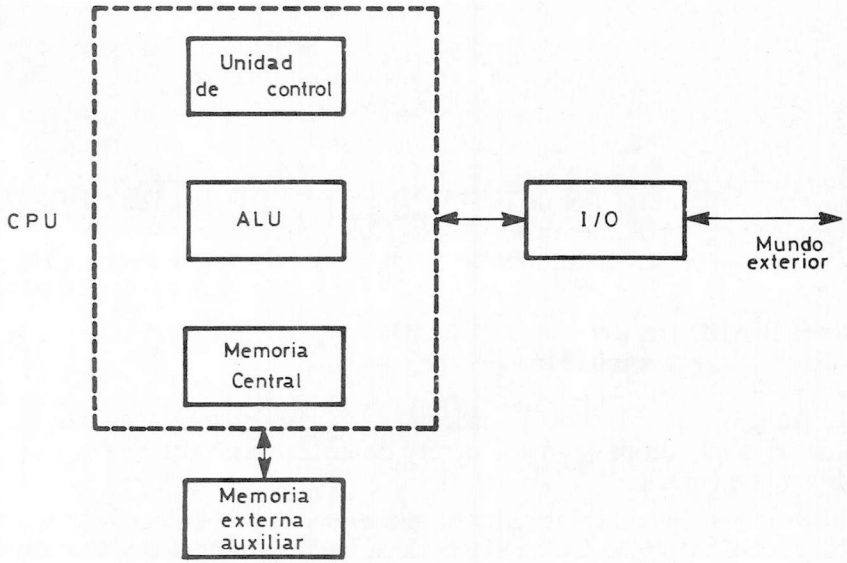


Fig. 3-1.–Estructura básica del ordenador.

nes y datos, estarán constituidos por 8 líneas y, por supuesto, al estar formada cada instrucción por 8 bits, el máximo número que de ellas puede existir será de $2^8 = 256$.

La necesidad de proporcionar al Decodificador de Instrucciones éstas en código binario, hace extremadamente complicado y engorroso escribir los programas usando dicho código. Un primer paso en la simplificación de la escritura de programas consiste en utilizar el código hexadecimal, con el que se consigue representar 4 bits con un solo dígito. A esta forma de representación de las instrucciones se la denomina *Código de Operación* (OP). Por ejemplo, una instrucción que en código binario o código máquina es 0110 1011, en hexadecimal (OP) sería 6B.

Como los números, tanto en código binario como en hexadecimal, no aclaran nada el objetivo de la instrucción que representan, se ha optado por utilizar expresiones relacionadas con el lenguaje humano. El lenguaje en términos NEMONICOS está muy relacionado con el lenguaje humano, puesto que emplea las abreviaturas de la función que realiza cada instrucción, derivadas del idioma inglés, para representarlas. Por ejemplo, la instrucción de "Cargar el Acumulador A" en inglés se escribe "Load A" y en código nemónico "LDA". Posteriormente, será necesario traducir a código máquina, que es el binario, los términos que expresan las instruc-

ciones en nemónicos, que junto con una serie de comandos auxiliares forman el lenguaje Ensamblador, para lo cual se precisará de un programa traductor que, en esta aplicación, recibe el nombre de programa "Ensamblador".

Las 3 formas de expresar las instrucciones expuestas hasta ahora: binario, hexadecimal y nemónico, están íntimamente ligadas con las características del Decodificador de Instrucciones. Existen lenguajes de más alto nivel, o con instrucciones más potentes, que son independientes de la máquina y que reciben el nombre de "compiladores". En estos lenguajes cada instrucción es un conjunto de instrucciones elementales, que admite el Decodificador.

Los lenguajes compiladores más comunes, son:

FORTRAN (Formula Translation). Aplicable a cálculos matemáticos.
COBOL (Common Bussines Oriented Language). Aplicable a la gestión de negocios.
BASIC (Beginner Algebraic Symbol Interpreter Compiler).
PL/1 (Programming Language One). Se utiliza para programación científica y comercial.

Para hacerse una idea de las diferencias entre trabajar con lenguajes de diferente nivel, se presenta a continuación el mismo programa, escrito en 4 lenguajes diferentes. Se trata de confeccionar un programa, que resuelva la siguiente ecuación: X = A + B – 22.

FORTRAN	ENSAMBLADOR	HEXADECIMAL	BINARIO
X = A + B – 22	LDA, A	00	0000 0000
	ADD, B	A	A
	SUB, 22	06	0000 0110
		B	B
		08	0000 1000
		22	0010 0010

Para obtener un programa en lenguaje binario o máquina (programa objeto), a partir de un programa escrito en un lenguaje de alto nivel, se precisa un programa traductor, que recibe el nombre de *"programa compilador"*. Las fases para llegar al *"programa objeto"* desde el *"programa fuente"*, escrito en lenguaje de alto nivel, se representan en la figura 3-2.

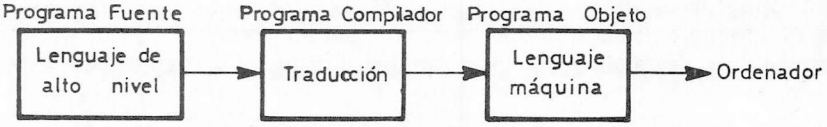


Fig. 3-2.–Obtención de un programa objeto desde un programa fuente.

FORMATOS Y MODOS DE DIRECCIONADO DE LAS INSTRUCCIONES

Se llama "formato" de una instrucción a la longitud que ocupa la misma y al cometido de las partes o bytes, que la componen. El modo de direccionado de una instrucción es la forma con la que se identifica la situación de los operandos que participan en la misma y el lugar donde se deposita el resultado. Como estas dos expresiones están íntimamente ligadas entre sí, se tratan conjuntamente.

Al referirse al "modo de direccionamiento" las instrucciones pueden dividirse en dos grandes grupos: 1) instrucciones que hacen referencia a memoria y 2) instrucciones que no hacen referencia a memoria. En las correspondientes al primer grupo, para su realización se requiere la participación de la memoria, puesto que uno o más operandos residen en ella. En el segundo grupo, los operandos se ofrecen en la instrucción, directamente detrás del código OP, o bien están contenidos en los registros de trabajo de la UCP.

Aunque más adelante, al analizar el juego de instrucciones de UCP concretas, se ampliarán el número y los detalles de las diversas formas que existen de direccionamiento, se presentan a continuación las más representativas:

Direccionamiento absoluto: En el caso de que las instrucciones consten de 8 bits y que el bus de direccionamiento de memoria esté formado por 16 líneas, para posicionar hasta 2^{16} locaciones, una instrucción, que haga referencia a memoria, tendrá 24 bits, repartidos en 3 bytes. El primero indicará el código de operación, que ha de interpretar el Decodificador y los dos siguientes, la posición de memoria donde se encuentra el operando que ha de manejar la instrucción.

Direccionamiento paginado: Hay un caso particular en el cual una instrucción, que haga referencia a memoria, consta sólo de 2 bytes. El primero de los cuales sigue indicando el código OP. Por otra parte, considerando la memoria de estructura paginada, a imagen de un libro, se supone que está distribuida en 256 (2^8) páginas, de las que se elige una con

los 8 bits del byte más significativo, mientras que cada página dispone de 256 posiciones, que se seleccionan con el byte de menos significado. El direccionamiento paginado especifica una página constante, por lo que sólo precisa de los 8 bits de menos peso para seleccionar la posición en ella.

Direccionamiento inmediato: Cuando el operando, que interviene en la instrucción, se proporciona directamente a continuación del código OP.

Direccionamiento indexado: Para dar mayor flexibilidad al direccionamiento de los operandos, que intervienen en la instrucción, se reservan en el campo de operación algunos bits, denominados "decodificadores", que permiten que intervenga en el direccionamiento de la memoria el contenido de un registro auxiliar, llamado Indice que se suma o resta al offset, que acompaña al código OP.

Direccionamiento indirecto: El contenido de la dirección de memoria, que proporciona la instrucción, es a su vez otra dirección, en donde se halla el operando. Para determinar el direccionamiento indirecto, se suele destina un bit del campo del código de operación, que recibe el nombre de "indicador".

Al estudiar los Decodificadores específicos de diversos microprocesadores, se ampliarán detalles sobre los variados modos de direccionamiento que dispone cada uno.

A continuación, en la figura 3-3, se ofrece un resumen gráfico de los modos de direccionamiento expuestos y que, como se puede comprobar, hacen referencia al formato de las instrucciones, que pueden constar de 2 y 3 bytes. No se ha considerado un tipo de instrucciones, que llevan inherente el operando o no lo necesitan, como, por ejemplo, puede ser poner a 0 todos los bits del registro Acumulador, o desplazar en el mismo 1 bit a la derecha el contenido de sus posiciones y que constan de 1 byte.

Los modos de direccionamiento expuestos, corresponden a los clásicos que utilizan los microprocesadores y se denominan "de una sola dirección", porque en todas las instrucciones sólo se puede realizar un acceso a memoria. Hay instrucciones, en otros tipos de ordenadores, "de 2 y 3 direcciones" con las cuales se puede acceder a varias posiciones de memoria para localizar datos o guardar resultados.

Así, por ejemplo, el procesador Univac del sistema 90/30, de los variados formatos que dispone para sus instrucciones, hay uno que detrás del código OP, define los dos operandos que intervienen en la instrucción, siendo uno de los mismos el destino del resultado. Dichos operandos o se suministran directamente, o se indica el registro de trabajo donde están contenidos, o la posición de memoria en la que se hallan almacenados.

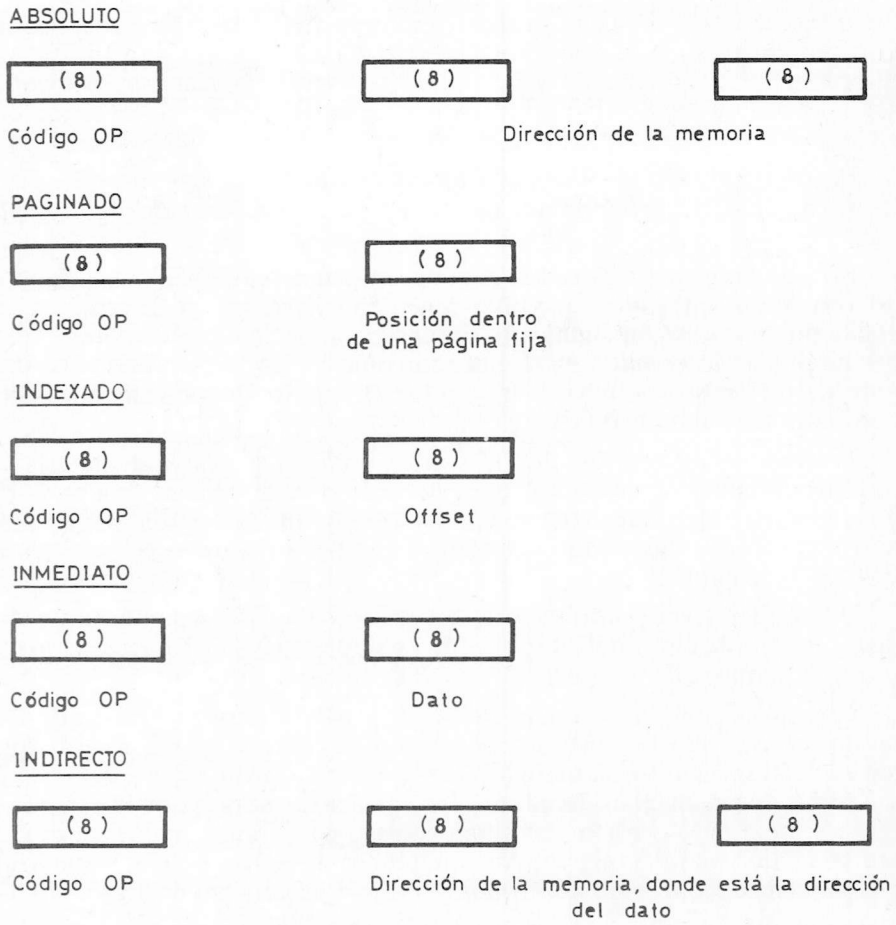


Fig. 3-3.–Formato de las instrucciones, según diferentes modos de direccionamiento.

CLASIFICACION DE LAS INSTRUCCIONES

En general, las instrucciones de cualquier tipo de ordenador se clasifican en los siguientes tipos, según el trabajo que realicen:

1) *Instrucciones aritméticas.* Encargadas de realizar las operaciones de tipo aritmético, que admite el procesador, por ejemplo, la suma.

2) *Instrucciones lógicas.* Efectúan operaciones de tipo lógico, por ejemplo, una operación AND.

3) *Instrucciones de lectura y escritura de memoria.* Se encargan de trasladar datos a/o desde las diferentes posiciones de memoria hasta cualquier registro de trabajo del ordenador, por ejemplo, la instrucción "Almacenar el Acumulador".

4) *Instrucciones de bifurcación.* Sirven para efectuar roturas en las secuencias del programa y son capaces de modificar el contenido del Contador de Programa, que en situaciones normales se incrementa una unidad cada vez que se realiza una instrucción para acceder a la siguiente.

   La bifurcación puede ser incondicional (se realiza siempre), o condicional. Como ejemplo de este último caso se cita una instrucción que permite la bifurcación (incremento o decremento del Contador de Programa en el número de posiciones que se desee), si el resultado de la operación que se ha realizado anteriormente ha sido negativo; en caso contrario la bifurcación no se realiza.

5) *Instrucciones de salto a subrutina y retorno.* Con este tipo de instrucciones también se cambia el contenido del PC, rompiendo la secuencia ordinaria del mismo y accediendo a zonas de programa repetitivas (subrutinas). La diferencia de estas instrucciones con las de bifurcación reside en que, antes de producirse el salto, en éstas, se guarda el estado del procesador y del PC en una parte de la memoria, para poder retornar al programa principal, una vez completada la subrutina.

6) *Instrucciones de transferencia entre registros.* Sirven para intercambiar los contenidos de los registros de trabajo del procesador. Dentro de este grupo pueden incluirse aquellas instrucciones que producen desplazamientos y rotaciones de los bits de los registros hacia ambos lados.

7) *Instrucciones especiales.* Entre ellas se encuentran las que originan la entrada o salida de los datos, instrucción de "paro" para detener el programa, de "no operar", que consume un tiempo sin producir ninguna variación y otras instrucciones específicas.

PROGRAMACION. DIAGRAMAS DE FLUJO

Conocido el juego de instrucciones de que dispone un ordenador, la confección de un programa de aplicación consiste en ordenarlas adecuadamente para ir resolviendo las sucesivas fases de que se compone el problema a resolver.

El primer objetivo en la programación de un ordenador digital debe ser, definir y comprender perfectamente el problema. Después, y teniendo en cuenta el juego de instrucciones del ordenador, o bien el lenguaje de alto nivel que se emplee, confeccionar el "diagrama de flujo", que ya se comentó. La siguiente operación consistirá en identificar cada una de las operaciones del diagrama de flujo con la instrucción correspondiente. Obtenido el programa en un lenguaje que acepte el ordenador, se carga en la memoria, se prueba y se depura, hasta que responda a plena satisfacción.

El ordinograma o diagrama de flujo, consiste en una descripción gráfica, paso a paso, del problema a resolver. Como se indicó, los símbolos más utilizados en los ordinogramas son los rectángulos y los rombos; los primeros representan la realización de una operación y los segundos la toma de una decisión. En la figura 3-4 se presentan los símbolos más frecuentes en los diagramas de flujo.

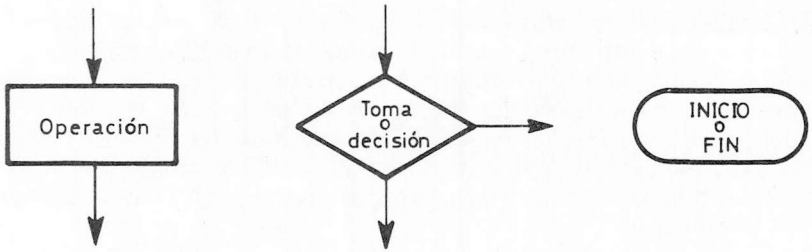


Fig. 3-4.–Símbolos frecuentes de los ordinogramas.

Para dar una idea de la estructura de un ordinograma, supongamos que se desea realizar uno, que sirva para efectuar multiplicaciones, teniendo en cuenta que el Decodificador de la UCP sólo dispone, como instrucciones aritméticas, las de la suma y la resta, en las cuales uno de los operandos está contenido en el Acumulador y el otro en una posición de memoria. El diagrama de flujo correspondiente se muestra en la figura 3-5. En él la expresión "almacenar" se refiere a llevar a la memoria el contenido del registro Acumulador y la de "cargar" significa que el contenido de una posición de la memoria se carga en el Acumulador.

La confección de un buen diagrama de flujo entraña un buen conocimiento de las instrucciones del ordenador, ya que una vez obtenido, hay que traducir cada bloque del diagrama a la instrucción o instrucciones que la implementen. Existen muchos caminos para resolver el mismo problema y, en la obtención del resultado más simple, influye tanto la experiencia del programador como los medios de que disponga para después depurar y mejorar el programa inicial.

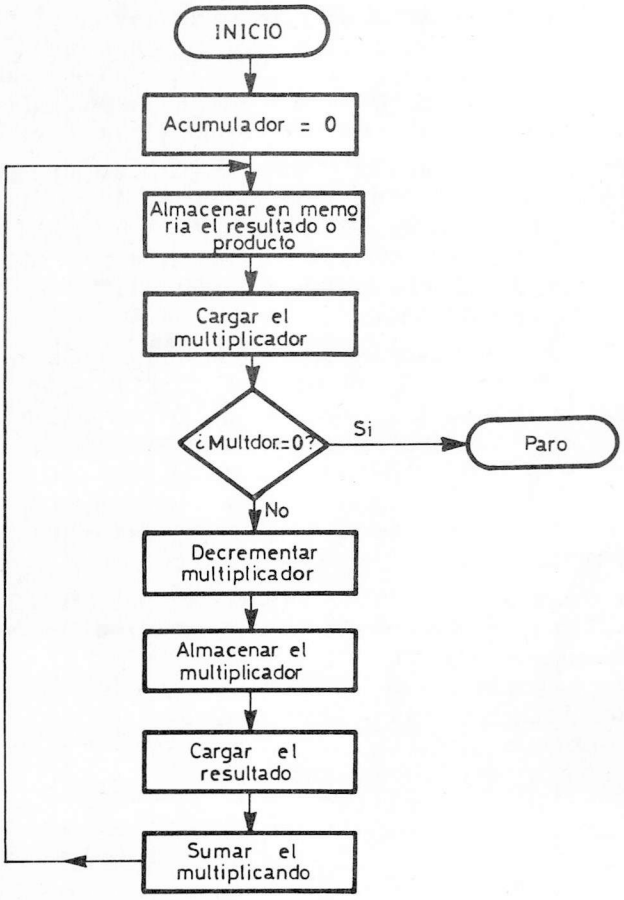


Fig. 3-5.–Ordinograma para realizar multiplicaciones a base de sumas.

De forma similar a como se describió en el capítulo anterior la arquitectura general de un ordenador, en el presente se expone un juego básico de instrucciones, sobre el que se soportará el software de un ordenador hipotético, de carácter didáctico y elemental.

JUEGO DE INSTRUCCIONES FUNDAMENTALES DE UN ORDENADOR

Para poder desarrollar algún programa, que aclare una serie de aspectos, que configuran el software del ordenador, se presenta en la tabla 3-1

un conjunto de las instrucciones más empleadas por casi todas las UCP. Su número se ha reducido al máximo, para abreviar esta exposición, que sólo trata de ofrecer una idea general del carácter y cometido de las instrucciones, para, posteriormente, comprender su empleo en la programación. Dicho juego de instrucciones no responde a la de ninguna UCP real.

En la primera columna de dicha tabla aparece el nombre de la instrucción. En la segunda viene la expresión nemónica de la instrucción, que se forma con sus abreviaturas, derivadas de su nombre en inglés. En la 3.ª y 4.ª columna se representa el Código de Operación en hexadecimal (OP) y en binario. La 5.ª columna indica el número de bytes o palabras de 8 bits, que forman la instrucción.

La simplificación del juego de instrucciones ha exigido que se emplee para la mayoría de ellas el direccionamiento absoluto, con el que se indica el operando mediante la dirección de memoria que lo contiene. También debe tenerse en cuenta que, en las instrucciones de bifurcación condicional o no, sólo se proporciona detrás del byte correspondiente al código OP otro byte, que actúa como offset, sumándose o restándose al valor del Contador de Programa, lo que significa que el mayor salto, que se puede conseguir con este tipo de instrucciones, es de 256 posiciones.

La última columna de la tabla 3-1 refleja, de una forma gráfica y esquematizada, el significado de la instrucción, teniendo en cuenta el significado de las siguientes abreviaturas:

A: Registro Acumulador.
(M): Contenido de la dirección de memoria M.
PC: Contador de Programa.
C: Acarreo o carry del Acumulador.
(E): Dato de entrada desde el exterior.
(S): Dato de salida.

JUEGO DE INSTRUCCIONES FUNDAMENTALES DE UN ORDENADOR

Instrucción	Nemónico	OP	Binario	N.º bytes	Expresión gráfica
A) *Instrucciones de transferencia con memoria*					
Cargar A	LDA	00	0000 0000	3	(M) → A
Almacenar A	STA	01	0000 0001	3	A → (M)
Intercambiar A	XMA	02	0000 0010	3	A ↔ (M)
B) *Instrucciones de Entrada y Salida de Datos*					
Entrada datos	INP	03	0000 0011	1	(E) → A
Salida datos	OUT	04	0000 0100	1	A → (S)

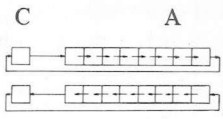
Instrucción	Nemónico	OP	Binario	N.º bytes	Expresión gráfica
C) *Instrucciones lógico-aritméticas con referencia a memoria*					
Complementar A	CMP	05	0000 0101	1	$A \rightarrow \overline{A}$
Sumar A	ADD	06	0000 0110	3	$A + (M) \rightarrow A, C$
Sumar A con acarreo	ADC	07	0000 0111	3	$A + (M) + C \rightarrow A, C$
Restar A	SUB	08	0000 1000	3	$A - (M) \rightarrow A, C$
Restar A con acarreo	SBC	09	0000 1001	3	$A - (M) - C \rightarrow A, C$
Operación OR	ORA	0A	0000 1010	3	$A \vee (M) \rightarrow A$
Operador EOR	EOR	0B	0000 1011	3	$A \veebar (M) \rightarrow A$
Operador AND	AND	0C	0000 1100	3	$A \cdot (M) \rightarrow A$
D) *Instrucciones de bifurcación condicional e incondicional*					
Bifurcación siempre	BRA	0D	0000 1101	2	PC ←PC ±(2.º Byte)/1/
Bifurcación si es mayor	BRP	0E	0000 1110	2	PC ←PC ±(2.º Byte) si resultado resta $> \phi$ (cero).
Bifurcación si es menor	BRN	0F	0000 1111	2	/1/, si resultado resta $< \phi$
Bifurcación si hay acarreo	BRC	10	0001 0000	2	/1/, si C = 1
Bifurcación si es cero	BRZ	11	0001 0001	2	/1/, si resultado de operación anterior $= \phi$.
E) *Instrucciones de salto a subrutina*					
Salto a subrutina	JMP	12	0001 0010	3	PC ←(2.º y 3.º Byte)
Retorno de subrutina	RTI	13	0001 0011	1	PC ←Posición desde la que se hizo el salto
F) *Instrucciones de desplazamiento*					
Rotación derecha con acarreo	RCR	14	0001 0100	1	
Rotación izq. con acarreo	RCL	15	0001 0101	1	
G) *Instrucciones inherentes de control*					
Borrar acumulador	CLA	16	0001 0110	1	$A \rightarrow \phi$
Paro/alto	HLT	17	0001 0111	1	Paro del programa
Incrementar A	INC	18	0001 1000	1	$A \leftarrow A + 1$
Decrementar A	DEC	19	0001 1001	1	$A \leftarrow A - 1$
No operar	NOP	1A	0001 1010	1	No hace nada.

Tabla 3-1.–Juego de instrucciones generales de un ordenador hipotético.

Las 27 instrucciones, descritas en la tabla 3-1, pueden considerarse fundamentales y comunes para todos los ordenadores reales.

Se recalca que las instrucciones de la aludida tabla sólo disponen de un modo para obtener el operando, pero muchas UCP pueden realizar la misma instrucción, utilizando diversos modos de direccionamiento, lo que aumenta notablemente su potencia.

EJEMPLOS DE PROGRAMACION INICIALES

Se ofrecen, a continuación, algunos ejemplos de carácter eminentemente didáctico, que porporcionen una idea sobre el manejo de instrucciones, en este caso, referidas a las de la tabla 3-1 del ordenador hipotético que se ha planteado.

1.º ejemplo

Un dato variable X se está almacenando continuamente en la posición de memoria 0030, otro dato variable Y en la 0031 y un dato, de valor fijo 22, en la posición 0032, para realizar la siguiente ecuación:

$$M = 2X + Y - 22$$

Si el resultado es negativo, éste se debe almacenar en la posición 0033 y detenerse el programa; en cambio si es cero o positivo, debe procederse a realizar un nuevo testado, puesto que X e Y pueden variar en cualquier momento.

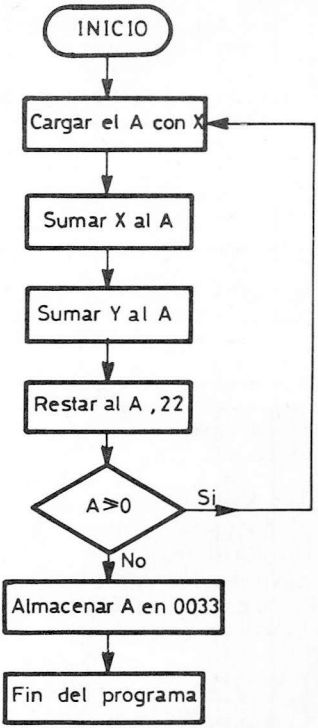
El organigrama de este sencillo ejemplo se muestra en la figura 3-6.

Al analizar el ordinograma de la figura 3-6, téngase presente que todas las operaciones aritmético-lógicas, que realiza la ALU, necesitan dos operandos, uno de los cuales siempre está contenido en el Acumulador y el otro, en general, en una posición de la memoria. El resultado de la operación se deposita en el Acumulador.

En la figura 3-7 se ofrece el programa correspondiente al ordinograma de la figura 3-6, en el que se ha procurado enlazar cada paso del ordinograma con la respectiva instrucción que lo efectúa.

Con la instrucción BRP hay que saltar 14 posiciones de memoria hacia atrás, con objeto de iniciar de nuevo el programa, en el caso de que el resultado M sea igual o mayor que cero. La instrucción consta de 2 bytes, el primero ofrece el código OP y el segundo el offset o salto que se debe

Fig. 3-6.–Organigrama correspondiente al 1.º ejemplo.



producir; como consta de 8 bits, el máximo será de 256 posiciones. Para distinguir el offset negativo, que significa un salto hacia atrás y en el que se resta al valor del PC el valor del offset, del offset positivo, o salto hacia adelante, el primero se expresa en la forma de complemento a 2, habitual para los números negativos. Si el offset 14 fuese positivo sería OE, que en binario es 0000 1110. El complemento a 1 de dicho número será 1111 0001. Sumando 1 al complemento a 1 se obtiene el complemento a 2: 1111 0010, que, en hexadecimal, es F2.

2.º ejemplo

Se trata de confeccionar un programa, que cumpla el ordinograma de la figura 3-5, destinado a realizar multiplicaciones a base de sumas. Arbitrariamente se elige la posición de memoria 0030 para contener al multiplicando, la 0031 para el multiplicador y la 0032 para el resultado o producto.

Dirección	Nemónico	OP	Binario	Comentario
0000 0001 0002	LDA 00 30	00 00 30	0000 0000 0000 0000 0011 0000	Cargar X, que está contenido en la posición 0030, en el Acumulador.
0003 0004 0005	ADD 00 30	06 00 30	0000 0110 0000 0000 0011 0000	Sumar X al Acumulador para obtener en él el valor 2X.
0006 0007 0008	ADD 00 31	06 00 31	0000 0110 0000 0000 0011 0001	Sumar el valor de Y, contenido en la posición 0031, al Acumulador.
0009 000A 000B	SUB 00 32	08 00 32	0000 1000 0000 0000 0011 0010	Restar al valor actual del Acumulador (2X + Y), el valor 22, situado en posic. 0032.
000C 000D	BRP F2	OE F2	0000 1110 1111 0010	Si el Acum (M) $\geqslant$ 0, salta el PC, 14 posiciones hacia atrás.
000E 000F 0010	STA 00 33	01 00 33	0000 0001 0000 0000 0011 0011	Si el resultado M $<$ 0 cargar el Acumulador en la posición 0033.
0011	HLT	17	0001 0111	Fin del programa
0030 0031 0032 0033	DATO X DATO Y VALOR 22 RESULTADO, SI ES NEGATIVO			

Figura 3-7. Programa correspondiente al ordinograma de la figura 3-6.

En la figura 3-8 se ofrece la solución de este ejemplo, situando en la primera columna y, al igual que en el anterior, la dirección de memoria para cada instrucción. Se ha comenzado arbitrariamente por la 0000. La segunda columna ofrece el símbolo nemónico de la instrucción y, si se trata de un dato o posición de memoria, su valor en hexadecimal. En la 3.ª columna (OP) se coloca el código hexadecimal de la instrucción, el valor del dato o la posición de memoria. La 4.ª columna ofrece las expresiones hexadecimales de la columna anterior, traducidas al código que entiende la máquina, o sea, al binario. Finalmente la 5.ª columna, detalla de una manera resumida el trabajo, que efectúa cada instrucción.

Dirección	Nemónico	OP	Binario	Comentario
0000	CLA	16	0011 0110	A ← 0
0001 0002 0003	STA 00 32	01 00 32	0000 0001 0000 0000 0011 0010	A → (0032)
0004 0005 0006	LDA 00 31	00 00 31	0000 0000 0000 0000 0011 0001	A ← (0031)
0007 0008	BRZ +12	11 0C	0001 0001 0000 1100	Si A = 0, saltar 12 posiciones adelante.
0009	DEC	19	0001 1001	A ← A − 1
000A 000B 000C	STA 00 31	01 00 31	0000 0001 0000 0000 0011 0001	A → (0031)
000D 000E 000F	LDA 00 32	00 00 32	0000 0000 0000 0000 0011 0010	A ← (0032)
0010 0011 0012	ADD 00 30	06 00 30	0000 0110 0000 0000 0011 0000	A ← A + (0030)
0013 0014	BRA EO	0D EO	0000 1101 1110 0000	Saltar hacia atrás 20 posiciones.
0015	HLT	17	0001 0111	Fin del programa.
0030 0031 0032	MULTIPLICANDO MULTIPLICADOR PRODUCTO O RESULTADO			

Figura 3-8.--Programa para realizar multiplicaciones a base de repetición de sumas, según el ordinograma de la figura 3-5.

3.º ejemplo

Se trata de sacar al exterior del sistema ordenador una onda cuadrada, en la que el nivel de salida pase alternativamente del valor 1 al 0. Se empleará la línea de menos peso del bus de datos, para sacar dicha onda. Para

conseguir el basculado en la línea de menos peso del bus de datos, se comienza cargando al bit de menos peso del Acumulador con un 0 y, una vez sacado al exterior, mediante la instrucción que realiza este cometido (OUT), se vuelve a cargar dicho bit con un 1, para sacarlo de nuevo al exterior. El incremento de una unidad a cualquier valor binario produce el cambio del valor del último bit, característica que se aprovecha para resolver este ejemplo, partiendo del ordinograma de la figura 3-9.

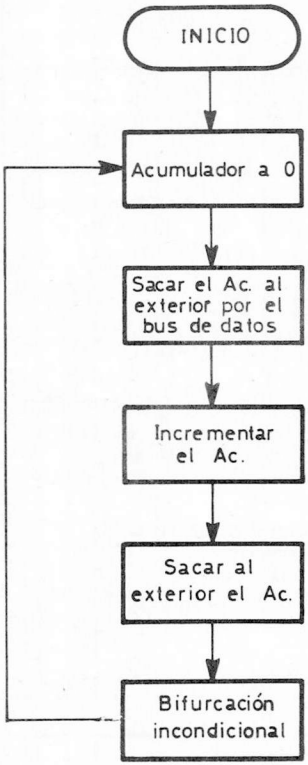


Fig. 3-9.—Ordinograma para obtener una onda cuadrada en el exterior. El programa a que responde este ejemplo se muestra en la figura 3-10.

El programa a que responde este ejemplo se muestra en la figura 3-10.

La frecuencia de la onda cuadrada dependerá de la duración de las instrucciones. Para disminuir la frecuencia de salida, será preciso perder un cierto tiempo entre la 2.ª y 3.ª instrucción, así como entre la 3.ª y la 4.ª Esta pérdida de tiempo puede lograrse y controlarse, introduciendo entre dichas instrucciones, que producen el basculado del Acumulador, un número adecuado de instrucciones NOP, que, sin modificar el estado

Dirección	Nemónico	OP	Binario	Comentario
0000	CLA	16	0001 0110	A ← 0
0001	OUT	04	0000 0100	A → Bus datos → Exterior
0002	INC	18	0001 1000	A ← A + 1
0003	OUT	94	0000 0100	A → Bus datos → Exterior
0004 0005	BRA FA	OD FA	0000 1101 1111 1010	Salto incondicional 6 posiciones atrás.

Fig. 3-10.–Programa para obtener en el exterior una onda cuadrada.

del procesador, suponen un retraso de tiempo. Esta solución supondría la confección de un programa muy largo, dada la gran cantidad de instrucciones NOP que se requerirán, y una sustanciosa ocupación de parte de la memoria. Para evitar estos problemas, se confecciona, dentro del programa principal, una subrutina de "retardo", "temporización" o "delay", que con pocas instrucciones retrase el paso de una instrucción a la siguiente. El ordinograma elemental de una subrutina de "retardo" se muestra en la figura 3-11.

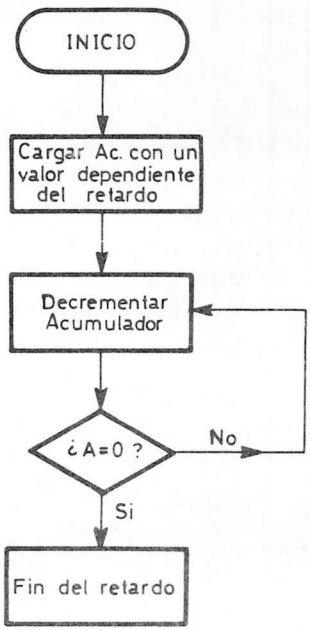


Fig. 3-11.–Ordinograma de un retardo o delay.

Para aumentar más la duración del retardo, se pueden unir en cascada varios bucles, hasta que se consiga el tiempo deseado.

En la figura 3-12 se presenta el programa, que responde al ordinograma de la figura 3-11, teniendo en cuenta que en la posición 0009 se carga el valor apropiado para lograr la temporización a conseguir (en este ejemplo se ha supuesto que es 05).

Dirección	Nemónico	OP	Binario	Comentario
0000 0001 0002	LDA 00 09	00 00 09	0000 0000 0000 0000 0000 1001	A ←(0009)
0003	DEC	19	0001 1001	A ←A − 1
0004 0005	BRZ 03	11 03	0001 0001 0000 0011	Si A = 0, saltar 3 posiciones hacia adelante.
0006 0007	BRA FB	OD FB	0000 1101 1111 1011	Salto incondicional de 5 posiciones atrás.
0008	HLT	17	0001 0111	Fin del programa.
0009	05	05	0000 0101	Valor según el retardo.

Fig. 3-12.–Programa para la realización de un retardo o temporización.

Si se supone que cada posición de la memoria del programa se ejecuta en 1 microsegundo (μs) (esta suposición es incorrecta, como se comprenderá al estudiar distintas UCP) y, como el bucle que hay en el programa de retardo, comprende 5 posiciones de la memoria desde la 3 a la 7 inclusive, dicho bucle se realizará en 5 μs. Dado que el dato cargado en la posición 0009 es 05, se recorrerá el bucle 5 veces, que representan un total de 25 μs. El programa completo consta, además del bucle, de las 3 primeras posiciones y la de paro, o sea, 4 μs más, que dan un total de 29 μs.

CAPITULO 4

El microprocesador

PRINCIPIOS Y GENERADORES

Conocidas las características esenciales, que reunen todos los ordenadores para introducirse con más rigor en las diversas categorías existentes, se comienza analizando los sistemas mínimos de tratamiento de la información, es decir, los menos potentes, que, a su vez, son los más modernos y se estructuran alrededor del "microprocesador".

Casi todos los bloques digitales, construidos con tecnología TTL (puertas, biestables, registros, memorias, etc.), también se fabrican en la actualidad con tecnología MOS. El reducido espacio necesario en la integración de componentes, junto con el bajo consumo, hacen idóneos a los circuitos integrados MOS para la implementación de circuitos muy complejos y repetitivos, con un precio bajo y una aceptable velocidad de trabajo.

Han sido tales los logros alcanzados en altas cotas de densidad de integración con tecnología MOS, que, recientemente (1971), se ha podido integrar en un sólo chip el sistema principal para el procesamiento de datos, con la posibilidad de realizar diferentes tipos de operaciones encadenadas, según una secuencia de instrucciones. Este elemento es la Unidad Central de Proceso, UCP, llamado "microprocesador", cuando se construye de esta forma. El microprocesador es la parte más importante de un ordenador y, comúnmente, contiene la Unidad de Control, la Unidad Lógico-Aritmética y diferentes contadores y registros de trabajo.

Para el diseñador de circuitos digitales, la posibilidad de utilizar el microprocesador como un componente más, ha supuesto un cambio total en la realización de proyectos y en el empleo de materiales clásicos. Los técnicos electrónicos disponen ahora y deben saber incorporar este arma potentísima para desarrollar sus proyectos: "el sistema ordenador". Para su empleo se deberán aprender una serie de cuestiones, derivadas de la programación y que estructuran el "software". Por otra parte, los profesionales de la informática deben darse cuenta de las posibilidades ilimi-

tadas de aplicación de los sistemas de tratamiento de la información, con la llegada de este componente electrónico; pero, dada la sencillez del sistema, será necesario que comprendan la tecnología y la forma de implementar el circuito electrónico, para que aprovechen al máximo sus posibilidades.

La llegada del microprocesador vino determinada por el elevado nivel de complejidad, que habían alcanzado los diseños de autómatas secuenciales, en los que las salidas, no sólo dependen de las entradas (sistemas combinacionales), sino también de la historia del proceso, del estado anterior de las salidas, o bien de un programa interno almacenado. La resolución de este tipo de problemas, acompañado por el desarrollo de la tecnología de los circuitos integrados e impulsado por la carrera espacial, fueron, sin duda, quienes empujaron a buscar una solución, tomando como modelo la configuración de Von Newman, la cual, recuérdese, constaba de 4 partes:

1) *Generador del tiempo,* que proporciona las señales de reloj, según las cuales se producen sincrónicamente los procesos e intercambios de información digital.

2) *Memoria,* parte de ella está reservada para contener el programa de instrucciones, según el cual se procesan los datos, y otra parte, a los datos requeridos o que se van produciendo. Precisamente fue la aparición de memorias de gran capacidad en un sólo chip, lo que posibilitó la implementación práctica de la estructura, que se comenta.

3) *Unidad Central de Procesamiento,* UCP, o también CPU, formada por varios bloques digitales y a la que, genéricamente, se llama "microprocesador", cuando está fabricada en un chip. Esta unidad, decodifica las instrucciones almacenadas en la parte de memoria que contiene el programa, produce las señales que gobiernan al resto de los componentes del sistema y efectúa las operaciones lógicas y aritméticas.

4) *Elementos de entrada y salida,* denominados también, por derivación de las palabras inglesas, Input-Output "I/O". Se encargan de aceptar las entradas de información de los elementos exteriores y transmitirlas adecuadamente al sistema y viceversa.

La figura 4-1 muestra de forma resumida la estructura de Von Newman.

La estructura de Von Newman se puede aplicar a cualquier tipo de proyecto, siempre que su rapidez esté en función con la del proceso externo a controlar y exista una adecuada adaptación entre la misma y los periféricos.

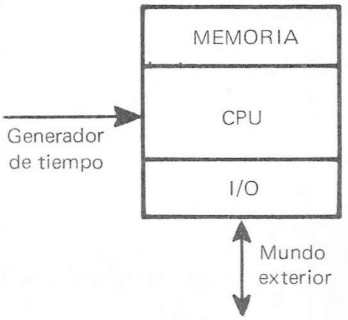


Fig. 4-1.–Estructura simplificada del sistema de Von Newman.

Las firmas pioneras en la comercialización de los microprocesadores, como componentes integrados sobre un chip, fueron en 1971 "Intel" con su modelo 4004 y "Rockwell" con el PPS-4, ambos de tecnología MOS de canal P; procesaban palabras de 4 bits, eran lentos, con instrucciones poco potentes y no admitían interrupciones.

La tecnología MOS de canal N se comenzó a utilizar en la segunda generación de microprocesadores, teniendo como ventaja fundamental sobre la anterior su velocidad, aunque a costa de ser menos estable a los ruidos y parásitos.

En la tercera generación, los microprocesadores han ido incorporando nuevas características, como la integración en un solo chip de todo el sistema (UCP, memorias, I/O y reloj), posibilidad de direccionar la memoria de diferentes modos, diversos tipos de interrupciones, mayor rapidez, acceso directo a la memoria, mayor potencia de las instrucciones, mayor capacidad de memoria e introducción de modernas tecnologías en la fabricación.

Al disponerse en la actualidad de los 4 bloques, que forman la estructura de Von Newman en circuitos integrados, el sistema completo, la mayor parte de las veces, vendrá compuesto por un circuito electrónico poco complejo.

ARQUITECTURA DEL MICROPROCESADOR

El chip que conforma al microprocesador o CPU, no puede realizar por sí mismo ningún tipo de procesamiento de la información, necesita la colaboración de otras unidades para llevar a cabo dicho trabajo.

En la figura 4-2 se presenta de forma relevante la arquitectura del microprocesador, pero también su conexionado con las memorias, las unidades de entrada y salida, así como la necesidad de disponer de alimenta-

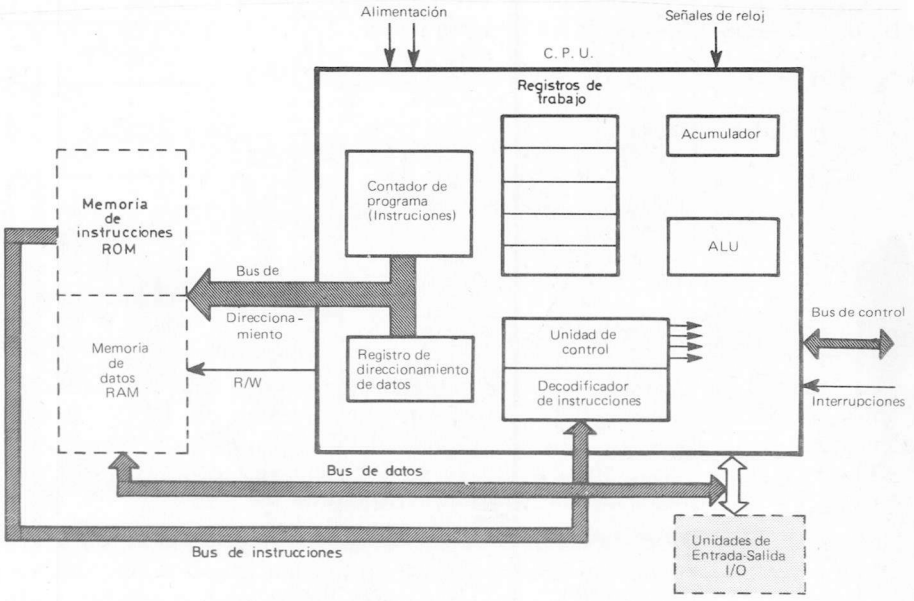


Fig. 4-2.–Arquitectura fundamental de la CPU o microprocesador y relación con el resto de los componentes del sistema. En general, el bus de datos y el de instrucciones forman un único bus bidireccional.

ción, señales de reloj y otras líneas auxiliares. La importancia del microprocesador estriba en reunir en una sola pastilla de circuito integrado a muchos componentes digitales, lo que permite construir la estructura fundamental de un ordenador con muy pocos elementos, generalmente, también circuitos integrados.

La figura 4-3 muestra un sistema ordenador, basado en un microprocesador, que se denomina "microcomputador". Se ha puesto con especial relieve en esta figura, la interrelación entre los diferentes bloques del sistema y la utilización común, por parte de todos ellos, de los tres buses de transferencias de información: el de *"direccionamiento"*, que, partiendo del microprocesador, envía en todo momento la dirección de la posición de la memoria, en donde se encuentra la instrucción o el dato que se deba procesar; el de *"datos"*, que transporta las instrucciones o los datos contenidos en la memoria e I/O, y, finalmente, el bus de *"control"*, por el que el microprocesador transmite y recibe señales, que controlan el funcionamiento de todo el sistema.

Del análisis de la figura 4-2 se desprenden los elementos, que forman parte de la CPU, entre los que destacan: La Unidad de Control con el De-

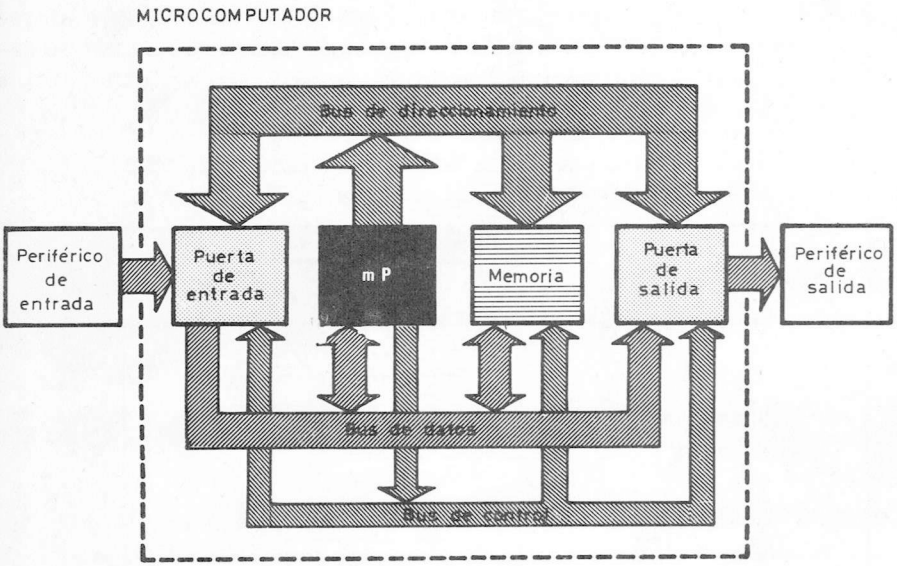


Fig. 4-3.–Arquitectura de un microcomputador basado en un microprocesador.

codificador de Instrucciones, la ALU, diversos registros de trabajo y contadores. La interconexión de la CPU con el resto del sistema se lleva a cabo por varios buses, o conjuntos de líneas: el de direccionamiento de la memoria, el de control, el de transmisión de datos y el de instrucciones. Estos dos últimos buses, normalmente, suelen ser uno único, como se aprecia en la figura 4-3.

El Contador de Programa (PC) proporciona, en cada momento, la dirección de la memoria, donde se encuentra la instrucción que hay que ejecutar. Dicha dirección sale desde la CPU por el bus de direccionamiento hasta la memoria; allí se selecciona la posición correcta y su contenido se transporta por el bus de instrucciones hasta el Decodificador, el cual interpreta el código recibido y ordena la realización de una serie de operaciones a base de la generación de un conjunto de señales desde la Unidad de Control, que se dirigen a los elementos, que deben actuar, a través de las líneas del bus de control. Por lo tanto, la Unidad de Control genera una secuencia de microórdenes, compuestas por una serie de impulsos sincronizados, que son conducidos a su destino por el bus de control, para llevar a cabo la instrucción decodificada.

Generalmente, en los microprocesadores las operaciones lógicas y aritméticas, que realiza la ALU, participan dos operandos, uno de éstos

proviene del registro Acumulador y el otro se recibe por un bus interno de entrada y salida de datos a la ALU. El resultado de la operación se cargará en el Acumulador y saldrá al exterior a través del mencionado bus interno, según queda representado en la figura 4-4.

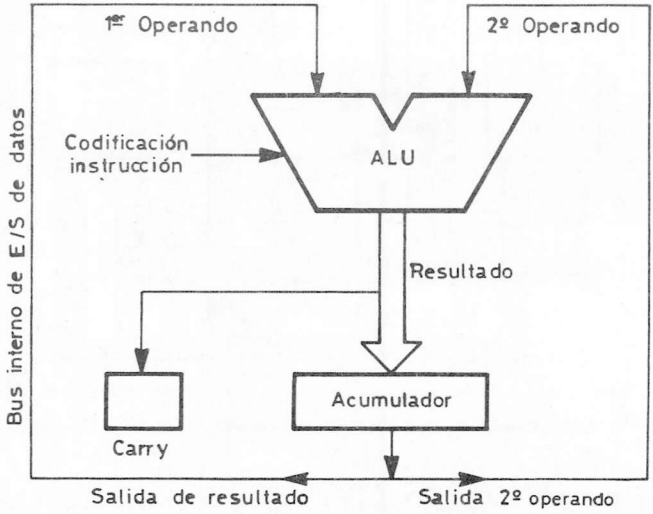


Fig. 4-4.–Conexionado y actuación de la ALU.

Cuando se ejecutan las instrucciones, pueden surgir resultados intermedios, que hay que almacenar temporalmente. Para ello se emplea una memoria, capaz de ser leída y escrita, es decir, del tipo RAM, a diferencia de la parte de la memoria en donde se guardan las instrucciones, que, por ser inalterables en los sistemas con microprocesador aplicados, es del tipo ROM, de sólo lectura.

Finalmente, es necesario introducir información desde el exterior, o bien sacar la que ya se ha procesado fuera del sistema, movimientos que se llevan a efecto a través del bus de datos y de los módulos de entrada y salida.

Todo el tratamiento de la información está sincronizado por señales de reloj de elevada frecuencia. También se debe proporcionar al sistema la alimentación que requiere. Entre las restantes señales auxiliares destacan las de interrupción, que detienen, en ciertos casos, el procesamiento de los datos y las de lectura-escritura, que determinan en la memoria RAM de datos la operación, que se ha de realizar en ella.

En resumen, la CPU, en conexión con la memoria de instrucciones (ROM), la memoria de datos (RAM) y las unidades de entrada y salida, forman un sistema mínimo de procesamiento de la información, utilizando como elemento prioritario de gran potencia al microprocesador (CPU integrada sobre un chip). La complejidad del circuito y el número necesario de chips dependerá, en gran parte, de la longitud del programa de instrucciones, que influye directamente sobre la capacidad de la memoria, así como de las necesidades de adaptación del sistema con el mundo exterior (periféricos).

La utilización del microprocesador permite apartarse de la idea de los grandes ordenadores de propósito general, cuyo volumen y precio sólo les hace interesantes en aplicaciones de cierta envergadura. El microprocesador posibilita el uso del tratamiento de la información en procesos muy específicos, a bajo precio, como puede ser, por ejemplo, el control de una cocina doméstica o la regulación del encendido del motor de un vehículo. En principio, el sistema básico, que controlará los dos ejemplos mencionados, puede ser el mismo y sólo diferirán en el programa de instrucciones, que se ha grabado para cada caso en la memoria ROM y los módulos de I/O de interface o adaptación con el exterior. En la figura 4-5 se ofrece un sistema basado en microprocesador para una concreta aplicación.

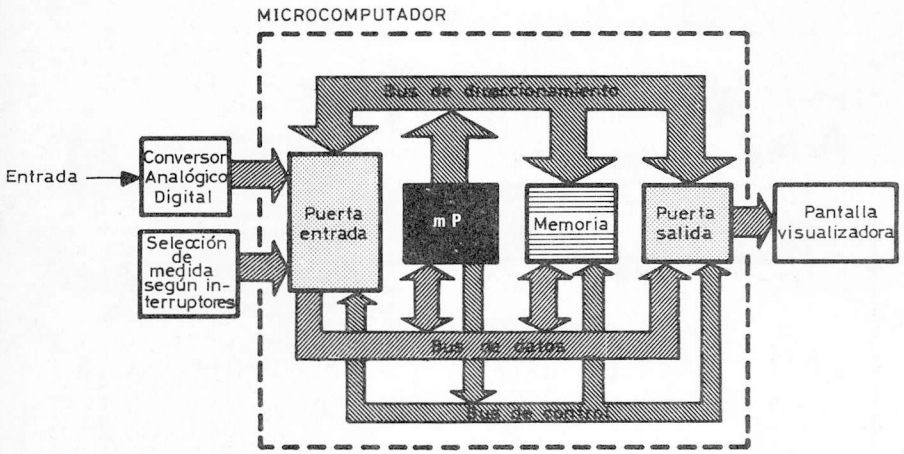


Fig. 4-5.–Sistema con microprocesador aplicado a un voltímetro digital.

La tendencia actual de los fabricantes es la de ofrecer en un solo chip el sistema completo, tal es el caso del 8748 de INTEL que, además de la CPU, contiene 64 bytes de memoria RAM, 1 K bytes de EPROM, así co-

mo puertas I/O, oscilador de clock y controlador de tiempos. Ver figura 4-6.

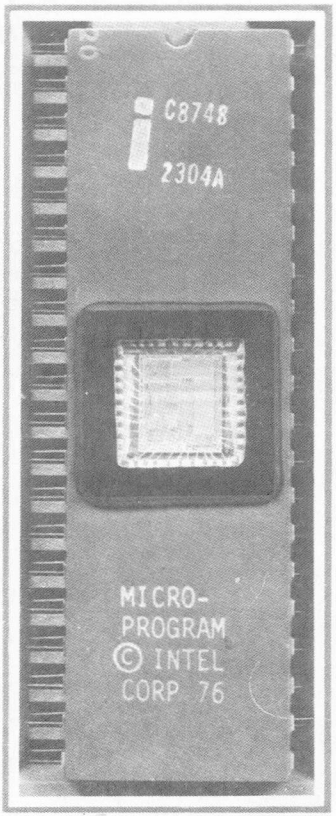


Fig. 4-6.–Foto del 8748, que contiene un sistema microcomputador completo.

ARQUITECTURA DE UN SISTEMA CON MICROPROCESADOR

El microprocesador o CPU no puede funcionar independientemente, necesita la colaboración de otros componentes para constituir un conjunto capaz de procesar la información.

La figura 4-7 muestra un conjunto básico con microprocesador y en ella se ha considerado que sólo existe una memoria, que contiene los datos y las instrucciones, que es direccionada por un bus de direccionamiento, que recoge la dirección codificada en binario desde el "Registro

de direcciones", que se carga desde el PC, cuando se trata de direccionar a una instrucción, o desde otro registro, cuando se direcciona una posición que contiene un dato.

Con relación a la figura 4-7, el contenido de cualquier posición de memoria pasa por el bus de datos hasta el "Registro de datos" y de allí se dirige al "Decodificador de instrucciones", caso de tratarse de una instrucción, o en el caso de tratarse de un dato, éste se procesa de acuerdo con las señales de control, que, partiendo de la "Unidad de control", se dirigen y gobiernan al resto de los elementos. Como se desprende de la figura 4-7 y por los conocimientos que ya se poseen, se deduce que las operaciones lógicas y aritméticas las ejecuta la ALU, junto con el Acumulador, y las informaciones, que proceden o se dirigen al exterior, pasan por las unidades de I/O.

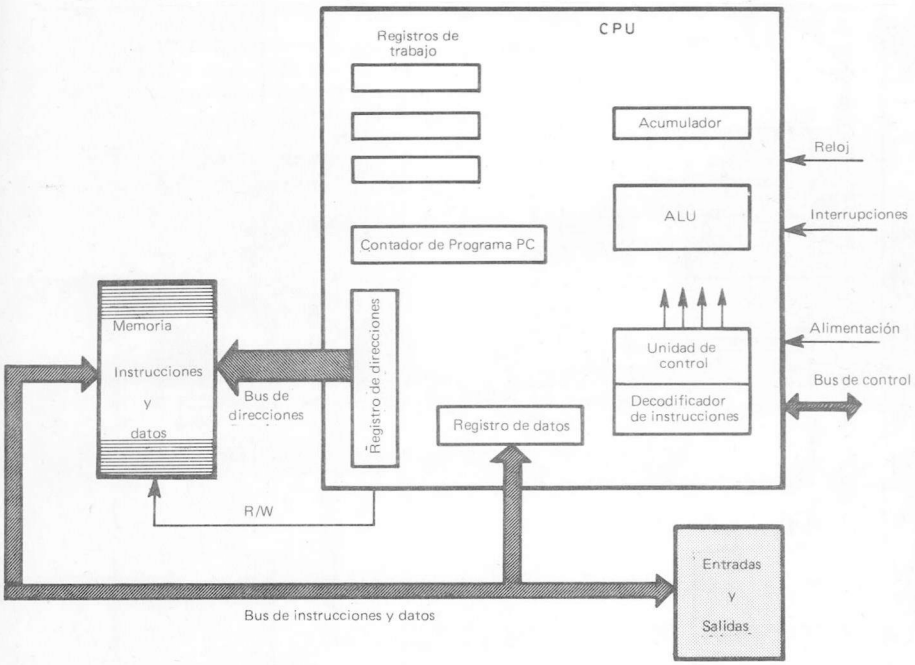


Fig. 4-7.–Conjunto básico de un sistema con microprocesador.

Por las líneas que parten de la Unidad de Control, a cuyo conjunto se denomina "bus de control", se transmiten informaciones y señales referentes a los contenidos y estados de los buses, a la determinación de la

lectura o escritura en la memoria, demanda y ejecución de interrupciones, conexión y desconexión de buses, sincronización interna y externa, impulsos de reloj y de inicio, etc.

La CPU, según la figura 4-7, queda interconectada al resto del sistema a través del bus de direccionamiento, bus de datos e instrucciones, bus de control y, finalmente, por las líneas de alimentación, de interrupción y de señales de reloj. Además de estos componentes fundamentales, cada fabricante dispone de diversos tipos de CPU, en los que existe una serie de registros de trabajo, que, según los casos, se emplean para direccionamiento, almacenamiento, intercambio, etc.

En la figura 4-8 se muestra una variante de la configuración clásica de un sistema con microprocesador, en la que merece la pena prestar atención a la transferencia de datos e instrucciones por buses diferentes.

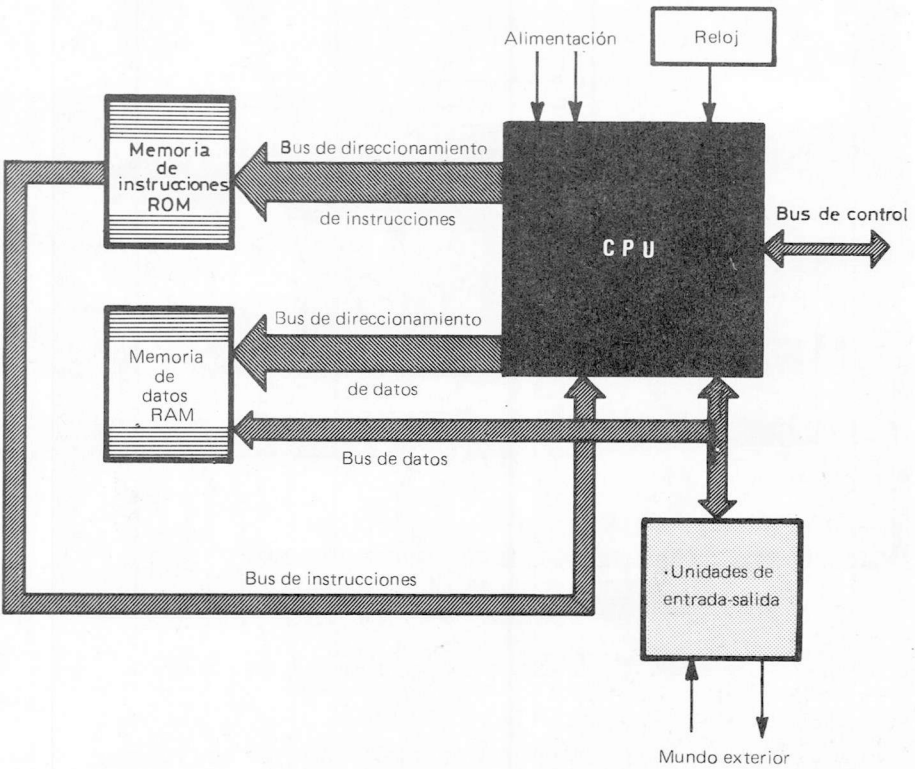


Fig. 4-8.–Variante del sistema general con microprocesador.

El sistema basado en un microprocesador, se ha dicho, tiene capacidad para realizar el tratamiento de la información, según el programa de instrucciones que almacena en su memoria, y su funcionamiento puede asemejarse al esquema de la oficina representada en la figura 4-9.

Los empleados que simbolizan los dispositivos de entrada y salida introducen al jefe de la oficina (CPU) los datos provenientes desde el exterior. El jefe procesa y trata los datos de acuerdo con unas instruccio-

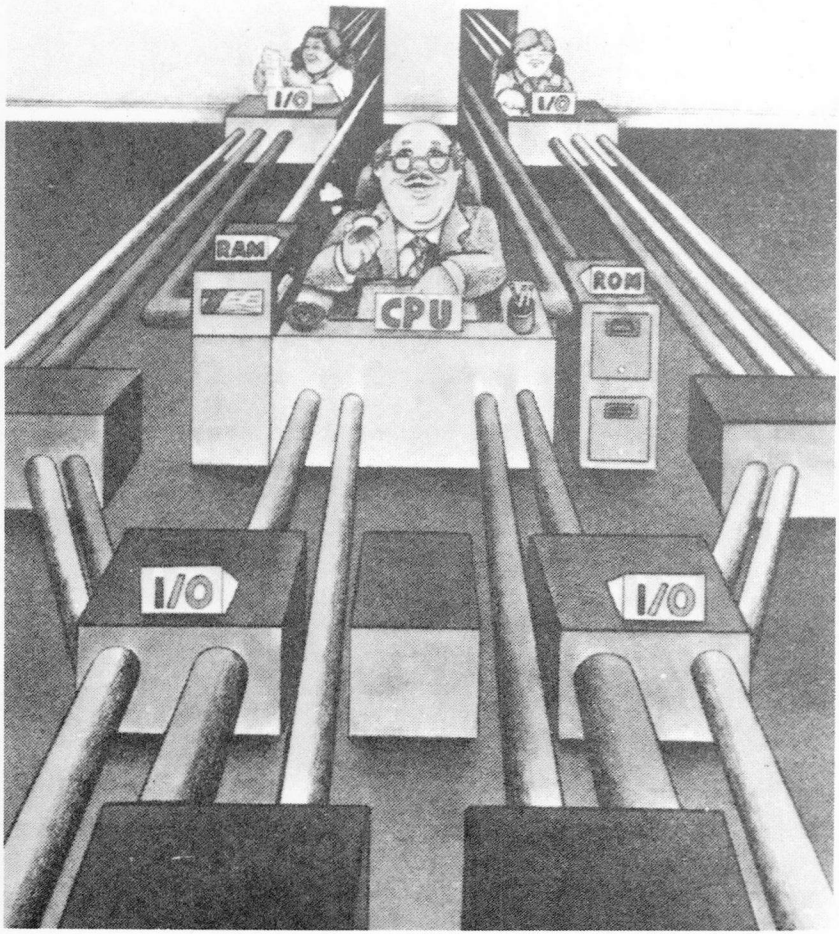


Fig. 4-9.–Similitud entre el trabajo de una oficina y el de un sistema con microprocesador.

nes fijas, guardadas en el archivo ROM (precios, plazos de entrega, condiciones especiales, etc.), mientras que en el archivo RAM guarda los datos y resultados provisionales, que surgen durante la realización de un proyecto. Finalmente, y una vez obtenidos los resultados, éstos se sacan al exterior a través de otros empleados I/O. Los diferentes puestos de la oficina están interconectados mediante los buses, por los que se transmiten los datos.

TRATAMIENTO DE LAS ENTRADAS Y SALIDAS DEL SISTEMA

La estructura de los sistemas con microprocesador, basada en los buses triestado, permite tres técnicas diferentes para el tratamiento de las entradas y salidas de datos.

1.ª) Tratamiento directo de las entradas/salidas desde la CPU mediante instrucciones específicas, tales como IN y OUT.

2.ª) Tratamiento de las entradas/salidas a través de interrupciones, que recibe la CPU desde los periféricos y que, en ciertos casos, es capaz de enmascarar y seleccionar.

3.ª) Las entradas y salidas son consideradas como posiciones de la memoria, lo que permite manejarlas con las mismas instrucciones y procedimientos que a la memoria. Aunque con este procedimiento disminuye la memoria útil y las instrucciones de entradas/salidas requieren 3 bytes para su completo direccionamiento, esta forma de tratamiento es la más usada, debido a su sencillez.

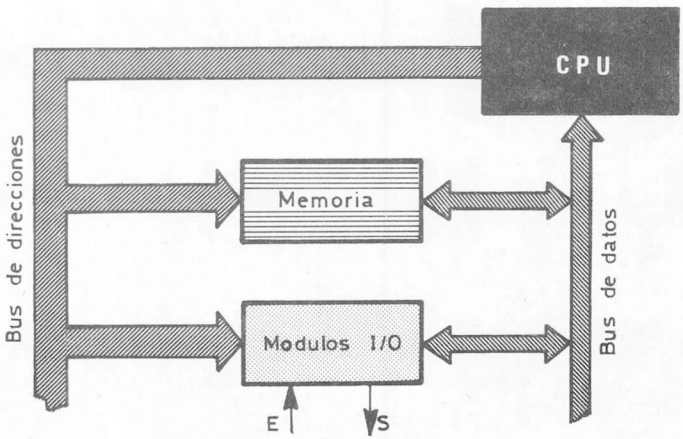


Fig. 4-10.–El control de la CPU sobre los módulos de I/O se realiza por el bus de direccionamiento.

Con el ejemplo específico que se analiza, se pretende demostrar el tratamiento que la CPU realiza generalmente sobre los módulos de entrada y salida, considerándolos como posiciones de memoria, que se pueden

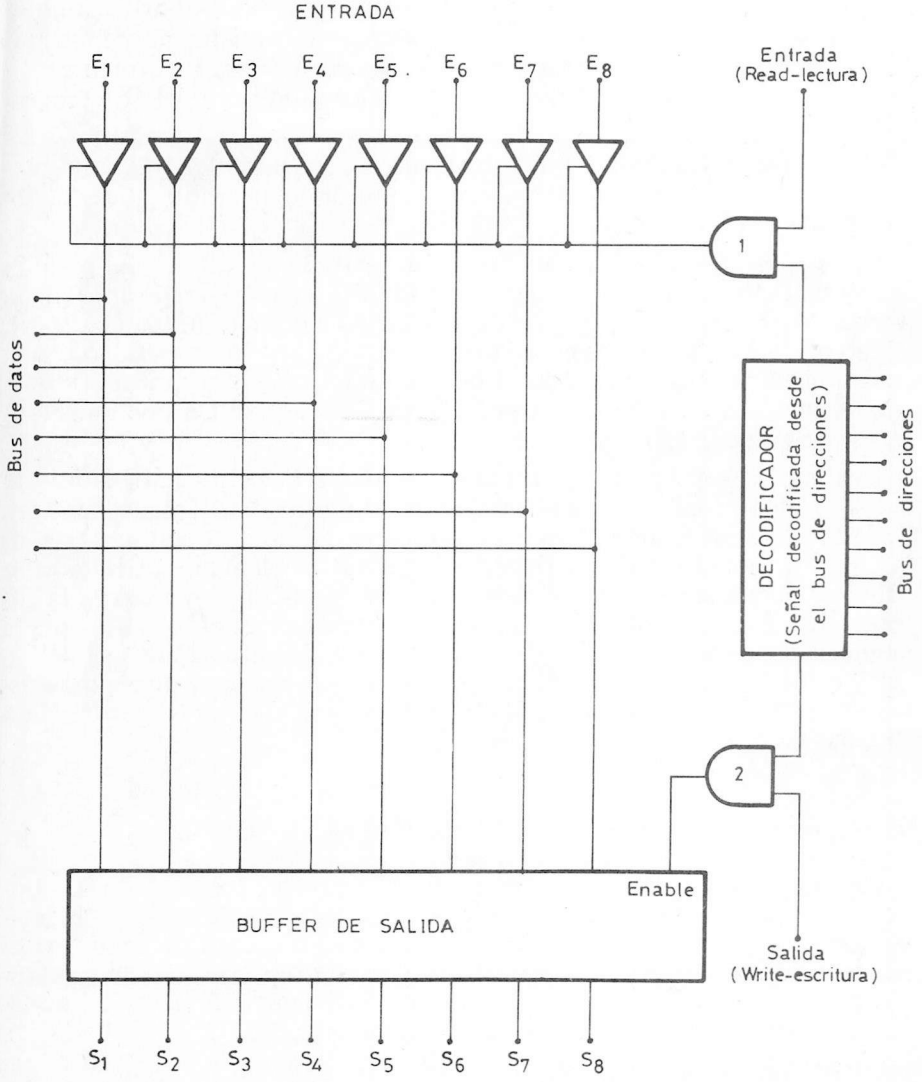


Fig. 4-11.–Esquema de un módulo de I/O específico.

leer o grabar. Así, una información de "entrada de datos" se traspasa desde el módulo de entradas y salidas (I/O) a la CPU (lectura de I/O), o bien, una salida de datos de la CPU puede enviarse al exterior a través del módulo I/O (escritura de I/O). Luego, la CPU controla a los módulos I/O mediante el bus de direccionamiento, al igual que a la memoria, como se ha esquematizado en la figura 4.10. A este tipo de tratamiento de los módulos de I/O por parte de la CPU, se denomina "mapa de memoria". También existen otras formas de tratar a los módulos de I/O, como el DMA.

Para aclarar el trabajo y funcionamiento de los módulos I/O se presenta en la figura 4-11 el esquema simplificado de una posible "puerta" de entradas y salidas de 8 bits.

En la figura 4-11 los niveles lógicos de entrada E_1, E_2, ..., E_8 pasarán al bus de datos para su tratamiento por la CPU, cuando quede activada la puerta AND 1, si recibe, además de la señal de lectura (entrada de datos), la dirección que se ha asignado a esta puerta, como si se tratase de una posición de memoria. Cuando el bus de direcciones transporta la dirección asignada a esta puerta I/O, el "decodificador" activa una de las entradas de las puertas 1 y 2.

Para que se produzca la entrada de los datos E_1, E_2, ..., E_8 al bus de datos, o bien la salida de la información del bus de datos a las salidas S_1, S_2, ..., S_8, es necesario recibir, en principio, la señal de entrada (Read-Lectura) o salida (Write-Escritura) para abrir los elementos triestado, o habilitar el buffer de salida de datos. De esta forma el contenido de la entrada (E_1, E_2, ..., E_8) pasa al bus de datos y, desde aquí, a la CPU o a la memoria de datos RAM, caso de tratarse de una lectura o "entrada de datos", o bien, el contenido del bus de datos, que proviene de algún registro de la CPU o de la memoria RAM, se transfiere al exterior por la salida (S_1, S_2, ..., S_8).

BUSQUEDA Y EJECUCION DE UNA INSTRUCCION

Con las ideas expuestas hasta este punto, se puede ya presentar la configuración básica de un sistema general con microprocesador y las relaciones internas y externas entre sus componentes, considerando que el tratamiento de los módulos de entrada y salida se maneja como posiciones de memoria direccionables, en las que se puede leer (entrada de datos) o escribir (salida de datos). Ver la figura 4-12.

Se explica, a continuación, el comportamiento de los elementos del sistema de la figura 4-12 para la realización de una típica instrucción, como lo puede ser una que lleve implícita una operación lógico-aritmética.

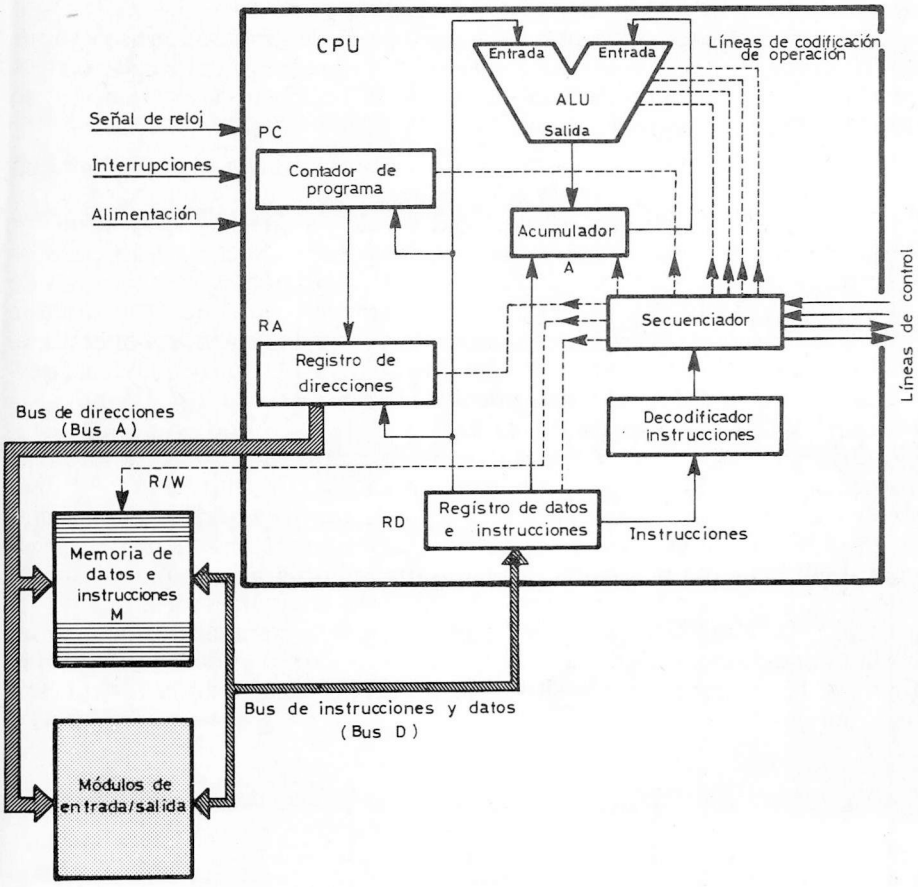


Fig. 4-12.–Sistema general con microprocesador e interrelación de sus componentes.

En principio, el Contador de Programa, que normalmente se incrementa con cada instrucción, a excepción de las que entrañan saltos o bifurcaciones, contendrá la dirección de la memoria del programa, donde reside la instrucción que se pretende realizar. El contenido del Contador del Programa pasa al Registro de direcciones y éste, a través del bus de direcciones, a la memoria (los módulos de entrada y salida también se determinan de esta forma, pero con diferentes direcciones). En la memoria su decodificador selecciona la posición, y su contenido, que será el código de la instrucción, se traslada por el bus de instrucciones y datos hasta el Registro de instrucciones y datos de la CPU.

Para concretar un poco más el desarrollo de la instrucción, se supone que se trata de una suma aritmética entre el contenido del Registro Acumulador y el contenido de una determinada posición de la memoria, en donde se guarda el otro sumando.

Siguiendo con el desarrollo de la instrucción, una vez el código máquina de la misma se encuentra en el Registro de datos e instrucciones, y por tratarse del código de una instrucción, de allí se envía al Decodificador de instrucciones, que la interpreta y actúa sobre el Secuenciador, el cual se encarga de proporcionar las órdenes de control, que cubren las fases de ejecución de la instrucción, que estará compuesta por una serie variable de microórdenes o microinstrucciones. Con esta instrucción concreta, el Secuenciador actuará sobre los elementos de la CPU para comenzar localizando en la memoria el dato, que actúa como uno de los sumandos: la composición de la instrucción de suma, además del código de la operación, dispondrá también de la dirección del sumando, que residirá en la siguiente o siguientes posiciones de la memoria. Decodificado el código de la operación, el Contador de Programa, que se habrá incrementado automáticamente, lanzará a través del bus de direccionamiento la dirección de la siguiente posición de la memoria del programa, donde radica la posición de la memoria de datos, en donde se encuentra un sumando (la posición de la memoria de datos completa puede abarcar dos posiciones de la memoria del programa si el bus de direcciones es de 16 bits). En la figura 4-13 se representa la situación de las posiciones de la instrucción de la suma.

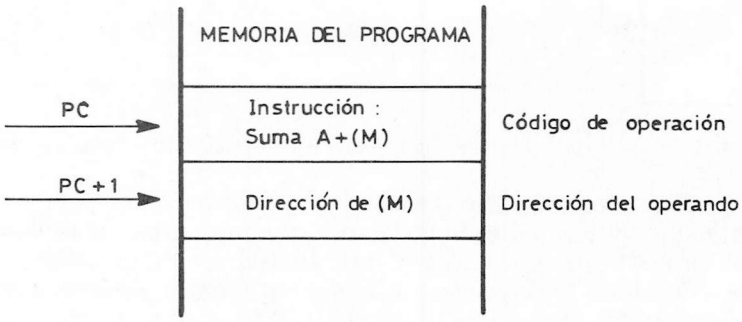


Fig. 4-13.—La introducción de suma está contenida en dos posiciones de la memoria del programa.

Recibida en la CPU la dirección de memoria, donde se encuentra el sumando, ésta se carga en el Registro de direcciones y por el bus de direccionamiento llega al Decodificador de la memoria. El sumando o dato sa-

le de la memoria por el bus de instrucciones y datos, hasta el Registro de instrucciones y datos. En este momento las señales de control, que se originan en el Secuenciador, completarán la fase de ejecución de esta instrucción. Para ello, una entrada de la ALU (recuérdese la figura 4-4) recibe el contenido del Acumulador y la otra el del Registro de instrucciones y datos. Asimismo, el Secuenciador aplicará a la ALU, por las líneas de codificación de la operación, las señales necesarias para llevar a cabo una suma aritmética. La ALU realizará la suma aritmética de las dos entradas y su resultado pasará de nuevo al Acumulador, con lo que quedará terminada la instrucción.

Como se deduce de este ejemplo, el Secuenciador una vez decodificado el código de operación, origina una serie de microórdenes, que permiten la ejecución de la misma. El número y actuación de estas microinstrucciones variará según los tipos de instrucción.

Un resumen simplificado de las fases de búsqueda y de ejecución de esta instrucción comporta una serie de movimientos y operaciones, que se representan a continuación, usando las abreviaturas indicadas en la figura 4-12.

$$PC \rightarrow RA \rightarrow Bus\ A \rightarrow M \rightarrow Bus\ D \rightarrow RD \rightarrow Decodificador\ Instrucciones$$

$$PC + 1 \rightarrow RA \rightarrow Bus\ A \rightarrow M \rightarrow Bus\ D \rightarrow RD \rightarrow [ALU = A + (M)] \rightarrow A$$

El "ciclo de instrucción" es el conjunto de operaciones, que completan las fases de búsqueda y ejecución de una instrucción; el "ciclo máquina" representa cada una de las operaciones elementales, que comprende el ciclo de instrucción. La señal de reloj, que suele recibir desde el exterior la CPU, divide al ciclo máquina en diferentes "estados". Para aclarar estos conceptos consideremos la instrucción de borrado, o puesta a cero, del Acumulador. Será bastante más simple que la de la suma aritmética y ocupará solamente una posición en la memoria del programa, pues no requiere ningún operando. De forma simplificada se representará:

$$PC \rightarrow RA \rightarrow Bus\ A \rightarrow M \rightarrow Bus\ D \rightarrow RD \rightarrow Decodificador\ Instruc. \rightarrow$$
$$\rightarrow Secuenciador \rightarrow [A \leftarrow O]$$

El "ciclo de instrucción" de esta instrucción de borrado del Acumulador también estará compuesto de las fases de búsqueda y de ejecución, pero las microórdenes que genera el Secuenciador, o sea, los ciclos máquina serán menos que en el caso de la instrucción de la suma.

REALIZACION DE SISTEMAS DE MICROPROCESADOR

La mayor parte de los microprocesadores actuales, y más en concreto su Decodificador de instrucciones, son capaces de interpretar un conjunto (set) de instrucciones, normalmente de número inferior a 100. Cada una de estas instrucciones se interpreta por el Decodificador de Instrucciones, que dispone grabado en memoria ROM los códigos de las diferentes instrucciones y compara y decodifica cada una, y el Secuenciador, que proporciona las órdenes adecuadas para la ejecución de la instrucción.

Las instrucciones se han de proporcionar al Decodificador en código máquina (binario), que es el único que puede interpretar. Por otra parte, las instrucciones deben ordenarse en forma secuencial de forma que el tratamiento que proporciona a los datos, sea el preciso para resolver y gobernar correctamente el proceso a que se aplica, es decir, el proyecto a resolver con el sistema a base de microprocesador.

Una vez establecida la secuencia específica de instrucciones, aplicable a un caso concreto, dicha secuencia, llamada "programa", se graba en fábrica sobre una memoria ROM, si la serie de sistemas a fabricar es superior a un número considerable de unidades, puesto que, al ser caro el proceso de grabación (alrededor de unos 5.000$), cuanto mayor sea la serie, más económica resultará la memoria. En los casos de series cortas o prototipos, el programa se graba directamente por el usuario o por una oficina especializada sobre una memoria PROM o REPROM.

Grabada la memoria de instrucciones, se colocan todos los componentes del sistema con microprocesador, soldados sobre una o varias placas de circuito impreso, cuyas pistas constituyen las líneas de los buses de enlace; el conexionado auxiliar procura la alimentación, la entrada de señales de reloj y las adaptaciones con el exterior.

Un sistema mínimo con microprocesador puede aplicarse a diferentes casos y las principales diferencias entre ellos afectarán sólo a la capacidad de memoria y a los "interfaces" con los periféricos exteriores. El primer punto dependerá de la complejidad del programa y del volumen de datos a procesar y el segundo, del tipo de elementos externos por los que se introducen o sacan los datos (impresoras, teletipos, TRC, relés, señales analógicas de corriente y tensión, etc.). La solución de cada problema depende fundamentalmente del programa almacenado en la memoria de sólo lectura (Software) y muy poco del cableado de interconexión entre los componentes, que suele ser muy parecido (Hardware).

Este cambio radical en el diseño y resolución de proyectos, que supone el microprocesador, al considerarle como un elemento que junto con

otros pocos forma un sistema ordenador, debe ser absorbido por los técnicos electrónicos, que proceden del Hardware, mediante el estudio de lenguajes y programación. Igualmente, los técnicos en Informática o programadores tendrán que aprender en profundidad la tecnología, en la que se basa la Electrónica Digital, para diseñar por completo sistemas con microprocesador. "El microprocesador impone a los técnicos, que lo utilicen, un conocimiento amplio tanto del Software como del Hardware."

Respecto a los lenguajes usados con los microprocesadores, se puede aplicar todo lo anteriormente referido a los ordenadores en general, aunque, dada la limitación en la aplicación de los mismos, el nivel de los lenguajes con los que se trabaja es bajo. El lenguaje Ensamblador, formado por los nemónicos de las instrucciones y un conjunto de reglas y comandos auxiliares, es el más utilizado, aunque es frecuente usar directamente los códigos OP, en hexadecimal e incluso en binario, en aplicaciones básicas. Sin embargo, la tendencia actual es la de ir aumentando el nivel de los lenguajes de alto nivel, aplicables a ciertos tipos de microprocesadores, como el PL/M de Intel (8080), el PASCAL de Texas y el MPL de Motorola (6.800), que generalmente disponen de un compilador cruzado en Fortan.

APLICACIONES DEL MICROPROCESADOR

Hasta hace poco, el tratamiento por ordenador de muchos problemas resultaba prohibitivo por el gasto que suponía la máquina, que era difícilmente amortizable. Los grandes y pequeños ordenadores que existían, estaban destinados al tratamiento de un amplio abanico de casos, por eso recibían el nombre *"de propósito general"*, o bien en un área, lo más extensa posible, como puede ser la gestión administrativa, los cálculos científicos, etc. Con objeto de que dispusiesen de la flexibilidad necesaria para usarse en muchos casos, se les dotaba de una gran capacidad de memoria, la máxima operatividad y la mayor velocidad posible, con lo cual su coste se hacía muy elevado.

Desde el momento en que se pudo integrar en un sólo chip a la CPU o microprocesador, se pudieron diseñar, en combinación con otros chips elementales (memorias y unidades I/O), circuitos de tratamiento de datos muy simples, que ocupaban poco espacio y eran de bajo coste.

La mayoría de los sistemas con microprocesador hasta ahora se han utilizado en la sustitución de circuitos complejos de lógica digital y, a medida que se introduce su conocimiento y se abaratan los precios de los componentes, la gama de sus aplicaciones se va ampliando rápidamente. La introducción de un sistema con microprocesador en el control de una

máquina-herramienta, en una balanza de peso-precio o en un automóvil, no supone encarecer el producto en exceso y, sin embargo, las posibilidades, fiabilidad, flexibilidad, y bajo consumo se incrementan de manera espectacular. Por otra parte, el microprocesador provee a los equipos, en los que se incorpora, una importante ventaja, cual es la versatilidad del mismo, pues, en general, cualquier cambio en el proceso de tratamiento de la información sólo afectará al programa de instrucciones.

Se puede recomendar, por todo lo expuesto, el empleo de sistemas con microprocesador en aquellos casos, que reunan las siguientes características:

a) Grandes series de fabricación.
b) Programas no muy complejos.
c) Previsibles cambios del tratamiento de información.
d) Interesa que ocupen poco espacio y sean de bajo precio.
e) Requerimientos de velocidad y niveles de ruido compatibles.

Se destacan a continuación algunas de las aplicaciones más importantes, en las que se utilizan actualmente los sistemas con microprocesador:

1) Sustitución de circuitos lógicos complejos (de más de 40 chips aproximadamente).
2) Control de electrodomésticos (hornos microondas, lavadoras, aire acondicionado, etc.).
3) Cajas registradoras y contables.
4) Balanzas peso-precio.
5) Control de ascensores.
6) Control de tráfico y semáforos.
7) Control de grandes paneles simuladores.
8) Control sobre las características de motores de vehículos.
9) Cajas negras de camiones y taxímetros.
10) Juegos y aplicaciones en TV.
11) Control de procesos industriales.
12) Máquinas-herramientas.
13) Máquinas calculadoras.
14) Instrumentación.
15) Terminales de datos.
16) Computadoras personales.
17) Electromedicina.
18) Control de motores de c.c. y de c.a.
19) Sistemas de supervisión y control remoto.
20) Control de telémetro y fotómetro en máquinas de fotos.

En estas y otras muchas aplicaciones, una vez analizado el problema y obtenido el programa objeto, se comienza probando con un prototipo

experimental todas las posibilidades y situaciones del funcionamiento real. Para ello, el propio usuario graba el programa sobre una memoria PROM, luego implementa el sistema sobre una placa de circuito impreso (CPU, RAM, PROM e interfaces de adaptación). El prototipo se conecta con los periféricos, que suministran y reciben los datos procesados y se analiza el comportamiento del sistema en todos los casos y condiciones posibles. Si las pruebas son positivas y la cantidad de producción alcanza una cifra respetable, se manda grabar directamente al fabricante de la ROM el programa que deben contener, para, finalmente, preparar la cadena de montaje y de comprobaciones para su producción en serie.

Conviene tener en cuenta que un sistema con microprocesador está constituido, en general, por una placa de circuito impreso, formada por varios chips o pastillas entre los que destaca el microprocesador, capaz de procesar los datos que se reciben desde el exterior o grabados en memoria, de acuerdo con el programa de instrucciones grabado en los chips de memoria de forma similar a un ordenador. La tecnología electrónica se emplea para fabricar el sistema y la programación para gobernar su actuación, y ambas técnicas no se deben deslindar para obtener los resultados óptimos. Por este motivo este libro pretende orientar al lector en los dos aspectos simultáneamente: los circuitos y los programas.

Una parte sustancial de esta obra se destina a la descripción de proyectos y diseños, basados en sistemas con microprocesador, para proporcionar al lector la base suficiente para que pueda interpretar, apreciar posibilidades y diseñar ideas propias.

APENDICE

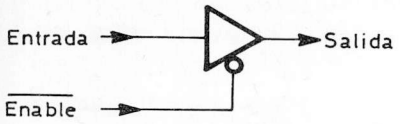
LOS ELEMENTOS TRIESTADOS EN LOS SISTEMAS CON MICROPROCESADOR

En el presente capítulo, al describir el comportamiento de un típico módulo para entradas y salidas, ya se ha comentado el trabajo de los elementos triestados. Dada la importancia de este tema y su continuo manejo en los sistemas con microprocesadores, a continuación se exponen de foma general las peculiaridades de estos elementos.

En los sistemas con microprocesador se puede observar que todos sus elementos intercambian información con el microprocesador a través del mismo bus o conjunto de líneas de interconexión. Haciendo referencia a la figura 4-2 y 4-3, por el mismo bus de datos puede circular una instrucción desde la memoria a la CPU en un momento determinado; como en otro, puede circular un dato desde el módulo de entrada, adaptado al periférico exterior, hasta la CPU. Precisamente este uso compartido de los buses es el que dota al sistema de una gran sencillez. El microprocesador se encarga de conectar un elemento del sistema al bus de datos para descargar en él la información, al mismo tiempo que debe desconectar a los restantes elementos del bus de datos, para evitar la producción de cortocircuitos, que invaliden toda la información. Lo que permite a la CPU seleccionar a un elemento para conectarle el bus de datos y desactivar a los restantes, es la capacidad de salida "triestado" de todos los elementos.

La figura 4-11 hacía referencia a buffers triestado. Un buffer, o separador, sirve para transmitir o dar paso a su señal de entrada hasta su salida, por eso también actúa como "driver" o conductor. El buffer tiene una salida especial, además de los niveles lógicos 1 y ϕ conocidos. A dicha salida se llama "flotante" o "de alta impedancia". Además de la entrada al buffer existe otra señal de control denominada $\overline{\text{Enable}}$ (activa por nivel bajo), que, cuando está a nivel bajo, el buffer transmite directamente la entrada hasta la salida, pero, cuando está a nivel alto, la salida queda en estado flotante o desconectada, independientemente del estado de la entrada. La figura 4-14 muestra el símbolo de un buffer y su correspondiente tabla de la verdad.

Para obtener una idea más clara del funcionamiento del buffer, se representa su funcionamiento en la figura 4-15, como si se tratase de un interruptor, gobernado por la activación de la bobina. Cuando $\overline{\text{Enable}}$ (Permiso) está a nivel 0, o tierra, circula corriente por la bobina y se cierra el contacto, con lo cual, en la salida se refleja el mismo estado que en la en-



$\overline{\text{Enable}}$	Entrada	Salida
0	0	0
0	1	1
1	0	flotante
1	1	flotante

0= bajo 1= alto

Fig. 4-14.–Símbolo y tabla de la verdad de un buffer triestado.

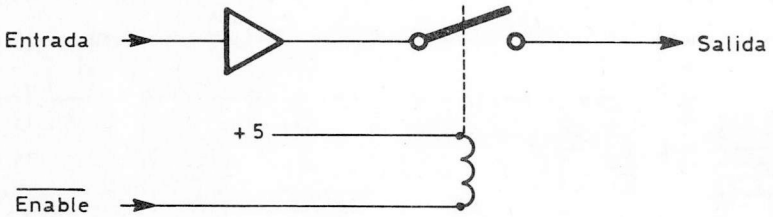


Fig. 4-15.–Funcionamiento equivalente de un buffer triestado.

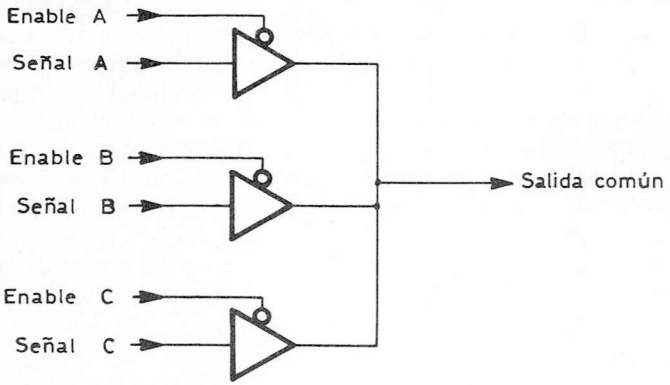


Fig. 4-16.–Compartición de una misma línea de salida por tres señales diferentes, mediante el uso de buffers triestado.

trada del buffer. Si $\overline{\text{Enable}}$ está a nivel 1, la bobina queda desactivada y el interruptor abierto, con lo cual, en la salida del buffer hay una "alta impedancia" o una desconexión.

Los elementos triestado son muy importantes, porque permiten que muchos componentes compartan una misma línea. La figura 4-16 ofrece el circuito de tres buffer triestado, que alimentan una misma salida. Sólo

uno de los tres buffers tendrá su señal $\overline{\text{Enable}}$ baja y él será el que proporcione el nivel de la salida igual al de su entrada. Los otros dos buffer estarán con salida en alta impedancia y, por lo tanto, no afectarán para nada la salida existente.

Muchos componentes, entre los que se encuentran los microprocesadores y las memorias, contienen buffers triestado internos. Estos circuitos integrados disponen de una línea, llamada Chip Select (CS: Selección de chip), o Chip Enable (CE: Permiso de chip), que controla las salidas de los buffers.

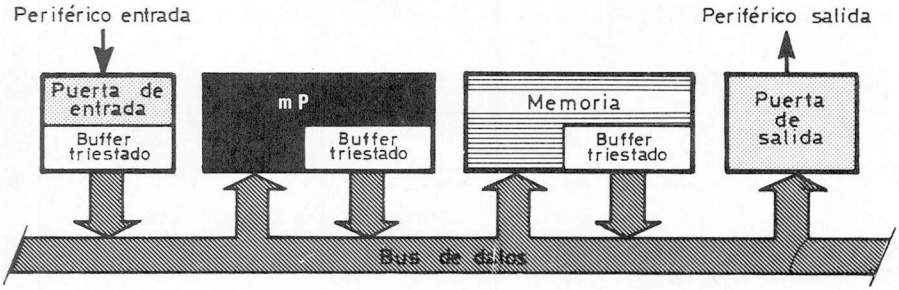


Fig. 4-17.–Actuación de los buffer triestado en los sistemas con microprocesador.

La figura 4-17 muestra cómo se usan los buffers triestado en los sistemas con microprocesador. Todos los elementos que ponen datos sobre el bus de datos tienen buffers triestado en sus salidas. El microprocesador genera señales de control (parte de las líneas del bus de control) para activar los buffer triestado de los elementos, desde los cuales se desea leer sus datos, mientras que los buffers triestado de los demás elementos quedan desactivados.

CAPITULO 5

Arquitectura y constitución de microprocesadores

INTRODUCCION

Sentadas las bases de los ordenadores en general y de los microprocesadores en particular por capítulos precedentes, se acomete ahora el estudio detallado de diversas versiones de microprocesadores existentes en el mercado. Se han elegido como los más representativos en la actualidad a nivel mundial los microprocesadores de 8 bits: R 6502, 6800 de Motorola y el 8080 y 8085 de Intel. En el presente capítulo se describirá la estructura, funcionamiento y aspectos relacionados con el hardware de los mismos. El siguiente capítulo se destina a exponer los temas relacionados con el software. Finalmente, existirá un capítulo dedicado a las familias de componentes, que completan el sistema a base de microprocesador. También se explica en los Apéndices de estos capítulos, el microprocesador 2650 de Signetics y el microordenador 8048.

Aunque cada fabricante de microprocesadores dispone de una arquitectura específica, la estructura básica se repite en todos ellos y, conociendo las 4 versiones que se comienzan a analizar, cuesta muy poco al diseñador, compenetrarse con cualquier otra, sobre todo, si tiene experiencia en trabajos con elementos digitales, puesto que se ha de intentar manejar al microprocesador como otro componente más.

1.ª PARTE: MICROPROCESADORES R6500

GENERALIDADES DE LOS MICROPROCESADORES R6500

La familia R6500 es un conjunto de chips diferentes, de aplicación en los sistemas microcomputadores, que se fabrican con tecnología NMOS y tratan palabras de 8 bits de longitud.

Dentro de la familia R6500 se incluyen 10 tipos diferentes de microprocesadores: R6502/3/4/5/6/7/12/13/14/ y 15. Los 4 últimos modelos son controlados por señales externas de reloj y hay dos tipos capaces de direccionar hasta 65 K de memoria: el R6502 y el R6512, los cuales son los únicos encapsulados con 40 pines, los restantes, con menos capacidad de direccionamiento, disponen de 28 pines.

La variedad de CPU o microprocesadores de esta familia, permite al proyectista la elección óptima del tipo que necesita, diferenciándose entre ellas, además del número de pines y líneas de direccionamiento de la memoria, por las señales de reloj y la cantidad de interrupciones disponible, como puede apreciarse en el cuadro comparativo de la figura 5-1. Los principales fabricantes de esta familia son MOS TECHNOLOGY y ROCKWELL.

CPU R6500

	40-PINES DIP		28-PINES DIP				
	R6502	R6512	R6503 R6513	R6504 R6514	R6505 R6515	R6506	R6507
Capac. direccionam.	65K	65K	4K	8K	4K	4K	8K
Interrup. Mascarable " No mascarable	Sí Sí	Sí Sí	Sí Sí	Sí No	Sí No	Sí No	No No
Línea de SYNC	Sí	Sí	No	No	No	No	No
Línea de RDY	Sí	Sí	No	No	Sí	No	Sí
Salida de Clock ϕ_1	Sí	Sí	No	No	No	Sí	No
Línea DBE	No	Sí	No	No	No	No	No

Fig. 5-1.–Cuadro comparativo de diferentes opciones de CPU de la familia R6500.

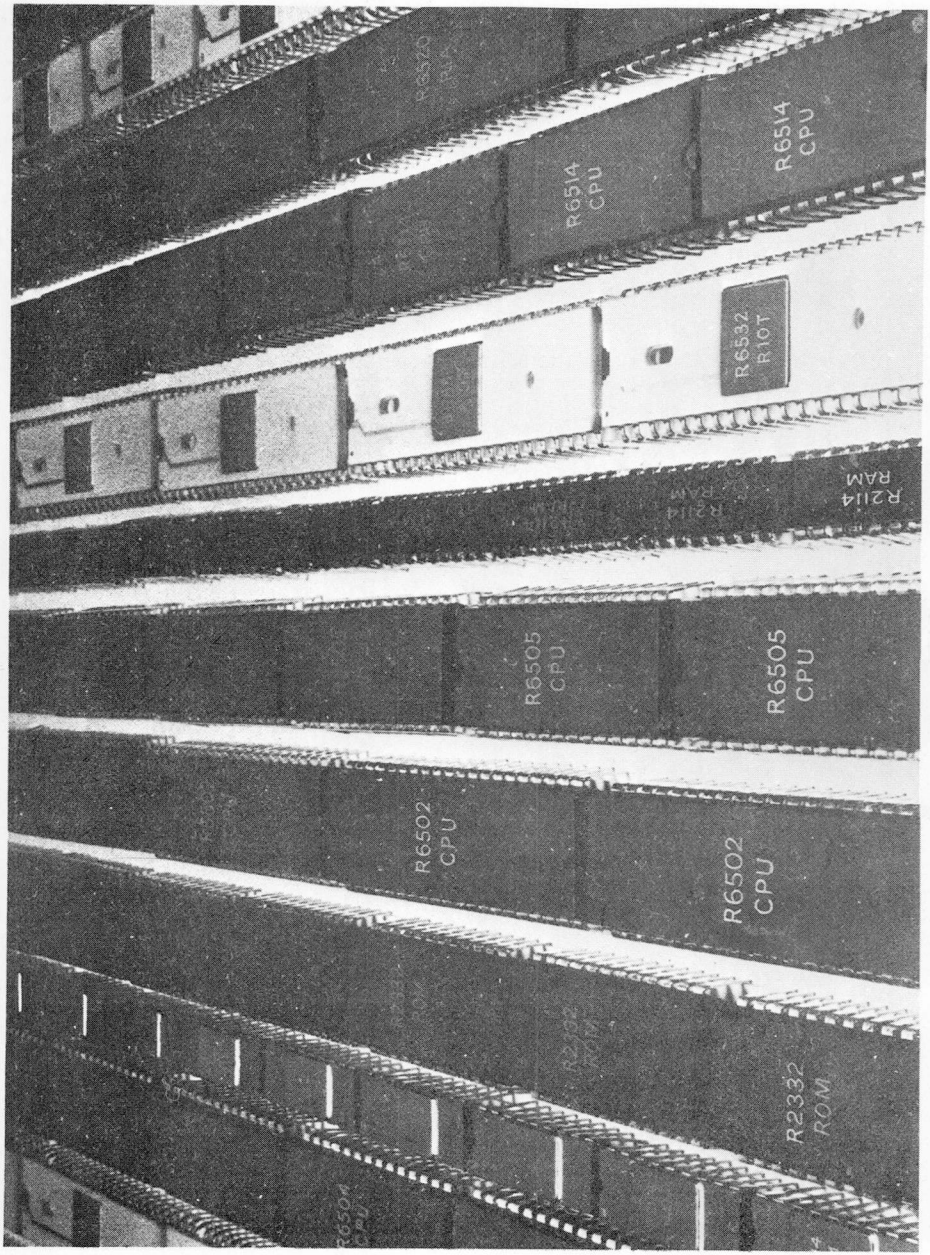


Fig. 5-2.–Fotografía de diferentes chips que forman parte de la familia R 6500.

Las características más destacadas de esta familia son:

- Unica alimentación de +5 V.
- Procesamiento en paralelo de palabras de 8 bits.
- 56 instrucciones, cada una de ellas con 13 formas posibles de direccionamiento.
- Aritmética binaria y decimal.
- Stack Pointer programable.
- Capacidad para interrupciones. Interrupción no enmascarable.
- Señal de reloj en el mismo chip o externa.
- Puede usarse con cualquier tipo y velocidad de memoria.
- Bus de datos bidireccional.
- Capacidad de memoria de hasta 65 K (65.536).
- Versiones de 40 y 28 pines y encapsulados plásticos y cerámicos.
- Frecuencias de trabajo de 1 MHz ó 2 Mhz, a elegir en todos los tipos.

Respecto a la última característica, que hace referencia a la velocidad del microprocesador, hay que tener en cuenta, que el fallo o mortandad de un chip integrado, depende de su frecuencia de trabajo, ya que cuanto más elevada sea ésta, mayor calentamiento provoca, así como de la cantidad de componentes que posea, puesto que es más probable que un diminuto grano de polvo se deposite en el chip durante su fabricación y estropee el conjunto. En otro orden de cosas, la mortandad junto con la cantidad de unidades que se fabriquen influyen directamente en el coste del chip.

En las versiones de CPU con frecuencia de trabajo de 1 MHz el tiempo de duración de un "ciclo máquina" es de 1 μs. La instrucción más rápida del sistema R 6500 se lleva a cabo en 2 ciclos máquina, luego tiene una duración de 2 μs. Para esta misma instrucción, pero con la versión de 2 MHz, en la que el ciclo máquina dura 500 ns, el tiempo que tarda en realizarse será 2×500 ns = 1 μs. La instrucción más larga es de 7 ciclos máquina, luego tendrá una duración de 7 μs con la versión de 1 MHz y de 3,5 μs con la de 2 MHz. Estos tiempos suponen una elevada velocidad de trabajo con respecto a otros microprocesadores, conseguida gracias a la estructura pipe-line, que permite realizar varias operaciones a la vez. La arquitectura pipe-line caracteriza a los sistemas ordenadores por poder iniciar las operaciones sin tener en cuenta la presencia de operaciones previas sin acabar. Estas velocidades son casi comparables con la de los microprocesadores construidos con tecnología bipolar, que, por lo general, son menos conocidos y aplicados.

ARQUITECTURA DE LA CPU DE LA FAMILIA R 6500

La arquitectura interna de todas las versiones de CPU existentes en la familia R 6500, es muy parecida a la del tipo general, comentada en el capítulo anterior y se muestra en la figura 5-3.

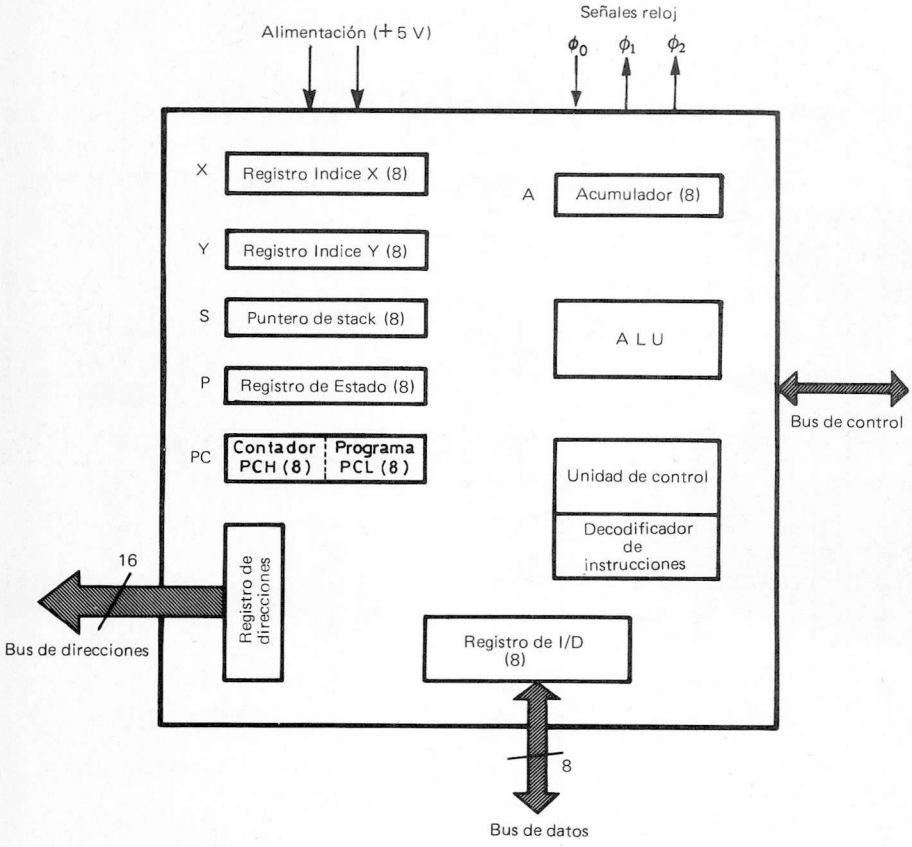


Fig. 5-3.–Constitución básica de la CPU de la familia R 6500.

Como se observa en la figura 5-3, la CPU sólo dispone de un bus de direccionamiento, común para la memoria que contiene los datos y para la de las instrucciones. Además de los elementos fundamentales, que posee toda CPU, tales como el Contador de Programa, Registros de direcciones, de instrucciones y datos, Unidad de Control, decodificador de instruccio-

nes, ALU y Acumulador, posee otros 4 registros de trabajo más, de 8 bits cada uno, que se detallan brevemente a continuación:

"Registros Indice X e Y": que realizan la función de Indice y posibilitan formas muy importantes de direccionamiento de la memoria.

"Registro de Estado, P": compuesto por 8 bits, denominados *flags* o *banderas* (indicadores), que determinan y señalizan el estado interno del sistema en cada momento.

"Puntero de pila": que guarda temporalmente el estado de los elementos y registros importantes en operaciones tales como los saltos a las subrutinas. Suele denominarse "Puntero de Stack" (Stack Pointer).

En la figura 5-4, se muestra la arquitectura interna de la CPU R 6502, esquematizada y simplificada, al objeto de apreciar fácilmente el interconexionado existente entre sus elementos.

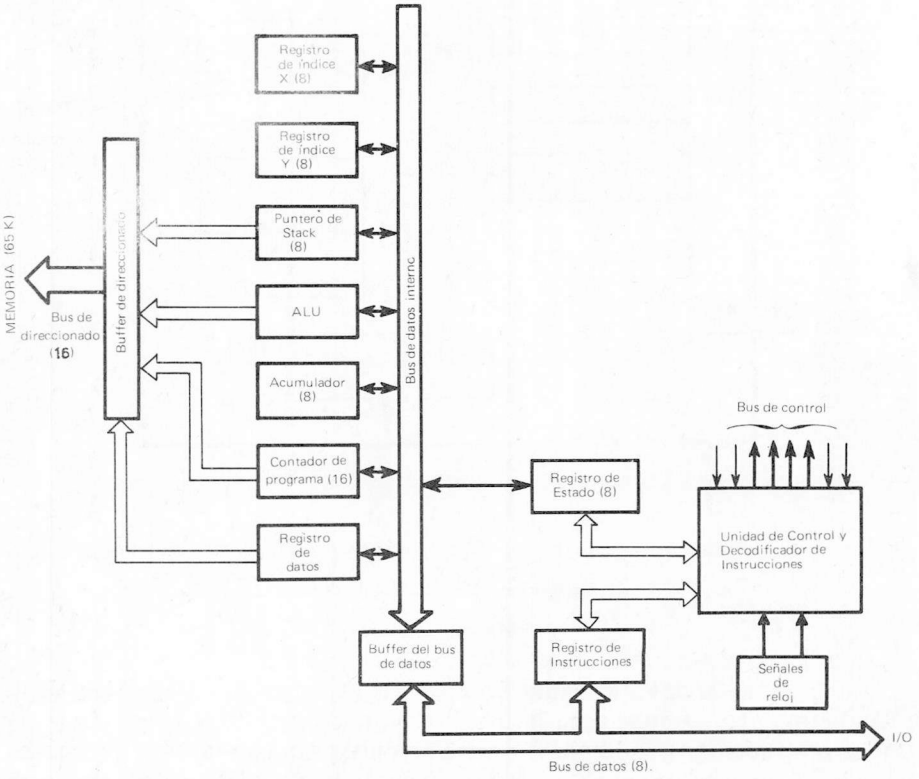


Fig. 5-4.–Esquema simplificado de la arquitectura interna de la CPU R 6502.

Una arquitectura más detallada de la CPU R 6502, se muestra en la figura 5-5, que sirve como base al resto de las CPU de la familia.

El Decodificador de instrucciones de la CPU de la familia R 6500 es capaz de entender e interpretar palabras de 8 bits, luego tiene la posibili-

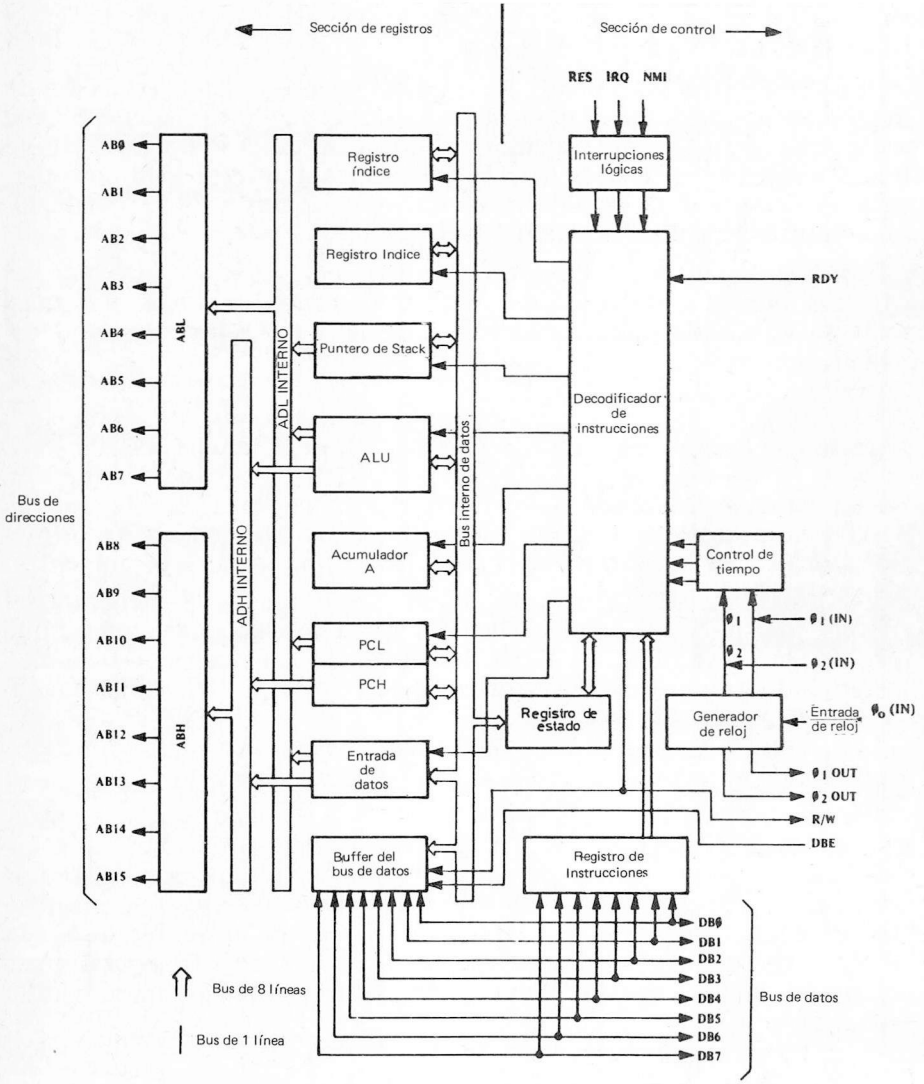


Fig. 5-5.–Detallada arquitectura interna de las CPU de la familia R 6500.

dad de aceptar hasta 256 instrucciones. No obstante, las CPU de la familia R 6500 sólo disponen de 56 instrucciones, que, al poseer cada una varios códigos, atendiendo al direccionamiento de los operandos, forman un total de 160 códigos de instrucción, dando aún un sobrante de unas 100 posibilidades. Téngase presente que la programación de un sistema con microprocesador es tanto más sencilla, cuantas menos instrucciones disponga y cuanto más potentes sean.

El bus de direccionamiento de la memoria posibilita la selección de una de sus posiciones y el número de las líneas que lo forman determina la capacidad máxima de la memoria. En el caso de la R 6502, el bus de direccionamiento consta de 16 líneas, con las que se pueden controlar 64 K posiciones de 8 bits, de memoria, equivalentes a 65.536 direcciones. (Refiriéndose a memorias, 1 K = 1.024.)

Para la entrada y salida de datos o de instrucciones existe otro bus con 8 líneas. Finalmente, existe un tercer bus, llamado *de control*, que transporta las señales de sincronización y gobierno a todos los chips, que forman el sistema.

"Función de los registros especiales de las CPU de la familia R 6500"

Aunque existen algunos elementos y registros en las CPU de la familia R 6500, cuya misión es conocida, hay otros, mostrados en las figuras precedentes, de los que aún no se tiene noticia y que pasamos a comentar:

Registros Indice X e Indice Y: Sirven para potenciar la estructura del software del microprocesador, dando lugar a instrucciones, en las que aparece una nueva forma de direccionamiento de la memoria. Del empleo de estos registros se derivan 6 formas de direccionamiento de la memoria: 1) Página cero, X, 2) Página cero, Y, 3) Absoluto, X, 4) Absoluto, Y, 5) Indirecto, X, y 6) Indirecto, Y. Todos los conceptos derivados de estos direccionamientos se expondrán en el capítulo siguiente.

Registro de Estado, P: Consta de 8 bits, que se simbolizan tal como se representa en la figura 5-6. Dichos bits reflejan algunas características importantes del estado interno del microprocesador tras la ejecución de una instrucción. Algunos de estos bits pueden ponerse por programa (mediante instrucciones), bien a 1 (set), o bien a 0 (reset); otros sólo disponen del estado resultante del procesamiento de los datos y otros pueden manejarse por ambos procedimientos. Además del interés intrínseco de los

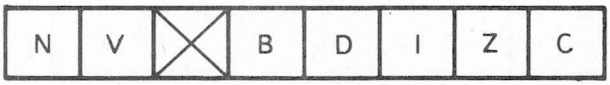


Fig. 5-6.– Símbolos representativos de los bits del Registro de Estado.

bits del registro de estado, de ellos se derivan unas instrucciones de salto condicional, que se producen o no, según el contenido de un bit de los 8, que forman este registro.

Los diversos bits de estado o flags, del registro de estado, expresan lo siguiente:

N: Flag de signo

El signo se representa, generalmente, mediante el octavo bit, de forma que, si tiene carácter positivo, tiene de valor 0 y, si es de carácter negativo, tiene de valor 1. Los restantes 7 bits indican el valor o magnitud.

En las operaciones aritméticas, el flag N tiene mucha importancia en la interpretación de los resultados.

V: Flag de sobrepasamiento u overflow

Cuando se efectúa una operación de suma de dos palabras de 8 bits, si se desea conocer el acarreo entre los dos séptimos bits, éste lo determina el flag V, que indica sobrepasamiento de la capacidad. Así, si el octavo bit es el de signo y se suman dos números de igual signo, tales como el +112 y el +45, el flag V quedará a 1, por dar una suma mayor a 127, la cual no puede expresarse únicamente con 7 bits.

Durante la realización de las instrucciones aritméticas de adición y substracción (ADC y SBC) el flag V representa el estado de sobrepasamiento de la capacidad, en la posición dedicada al signo (bit de más peso). El programador, que no emplea aritmética con bit de signo, puede ignorar totalmente este flag durante su trabajo; pero este flag tiene similar significado que el carry, cuando se usan números binarios con bit de signo.

C: Acarreo de los octavos bits

El flag C indica el acarreo, si lo hay, de los bits de más peso. Una operación de suma afectará a los flags V y C, de ahí su importancia en las operaciones aritméticas. Además de afectar a este flag el resultado de las operaciones aritméticas, puede alterarse su valor mediante instrucciones en el programa: la de borrado, o puesta a 0 (CLC) y la de puesta a 1 (SEC).

El bit C, de acarreo, se usa como noveno bit en las instrucciones de rotación y desplazamiento.

B: Flag Break

Este flag es puesto a 1 exclusivamente por la CPU, y determina durante una secuencia de un servicio de interrupción, si la misma ha sido ocasinada por una instrucción BRK, o bien por una interrupción real y exterior.

D: Flag para el tratamiento aritmético

Según el valor de este flag, el tratamiento aritmético de las operaciones que realiza la ALU, será de tipo decimal (BCD) o binario (hexadecimal). Para el tratamiento en modo decimal, D vale 1, lo que puede conseguirse fácilmente por programa, mediante la instrucción SED. La instrucción CLD borra o pone a 0 este flag, lo que implica el tratamiento binario de las operaciones.

I: Flag de interrupciones

Cuando este flag contiene un bit 0, permite la realización de interrupciones "mascarables" ($\overline{IRQ}$), mientras que, si vale 1, no lo permite y aisla al sistema de las interrupciones externas, como se explicará en breve. Este flag puede ponerse a 1 o a 0 por programa, mediante las instrucciones SEI y CLI.

Z: Flag cero de la operación anterior

Si en la operación anterior el resultado ha sido 0, es decir, todos los bits del Acumulador han quedado valiendo 0,Z pasa, sólo en este caso, a contener un 1.

Como aplicación de lo expuesto anteriormente respecto al Registro Stack Pointer, o Puntero de Stack, se ha de considerar que la memoria es tratada como de carácter paginado, lo que significa que sus diversas posiciones se subdividen en grupos de 256 posiciones, que reciben el nombre de "páginas". Las 256 posiciones de la página cero, ocupan las direcciones desde la 0 hasta la 255; la página 1 comprende desde la posición 256 hasta la 511 y así sucesivamente. Tanto la página 0, como la 1, son muy singulares y, en el caso de la última mencionada, su direccionamiento se hace a través del registro *stack pointer* (S), que es un contador de 8 bits, que sirve para direccionar las 256 posiciones de la página 1. El puntero del stack hace trabajar a la página 1 de la memoria como una pila tipo

LIFO o FIFO. Su principal misión es la de salvar a los registros importantes del microprocesador, cuando se efectúa un salto a subrutina, después del cual, se ha de regresar al programa primitivo. Así, por ejemplo, al inicializar un salto a una subrutina, si el puntero de stack señala una posición, en ella se carga el byte de menos peso del PC para que, a continuación, señale el stack la siguiente posición, en donde quedará almacenado el byte de más peso del PC. Al acabar la subrutina, el PC quedará repuesto con las posiciones salvadas anteriormente por el stack. También se salvan mediante el stack, los contenidos de ciertos registros de la CPU.

DIAGRAMA DE CONEXIONES DEL MICROPROCESADOR R6502

En la figura 5-7 se muestra la constitución simplificada de la CPU R 6502, a la izquierda y la configuración de sus pines a la derecha. El diagrama de la mencionada figura habrá que tenerlo muy en cuenta durante el conexionado de la CPU con el resto de los elementos del sistema.

La misión de los diferentes pines de entrada y salida del R 6502, que le relacionan con otros elementos del sistema con microprocesador, se describen a continuación:

Señales de reloj ϕ_1 y ϕ_2

El microprocesador R 6502 se suministra con un generador interno de señales de reloj, cuya frecuencia es controlada con un cristal exterior o con una red RC, que introducen una señal ϕ_0 por el pin 37, de la figura 5-7, obteniéndose la señal ϕ_2 por el pin 39, aplicada a circuitos integrados 7404, tal como queda representado en la figura 5-8. ϕ_1 es una señal inversa a ϕ_2.

Bus de direcciones ($AB_0 - AB_{15}$)

Son las 16 líneas de salida, que forman el bus de direccionamiento de la memoria. Estas salidas son compatibles con señales TTL. A la frecuencia de 1 MHZ la dirección es válida 300 ns durante la fase ϕ_1 de reloj y permanece estable hasta el próximo impulso ϕ_1.

Bus de datos ($DB_0 - DB_7$)

Hay 8 pines del R 6502, que constituyen el bus de datos. Se trata de un bus bidireccional, pues transfiere datos (e instrucciones) a la CPU y

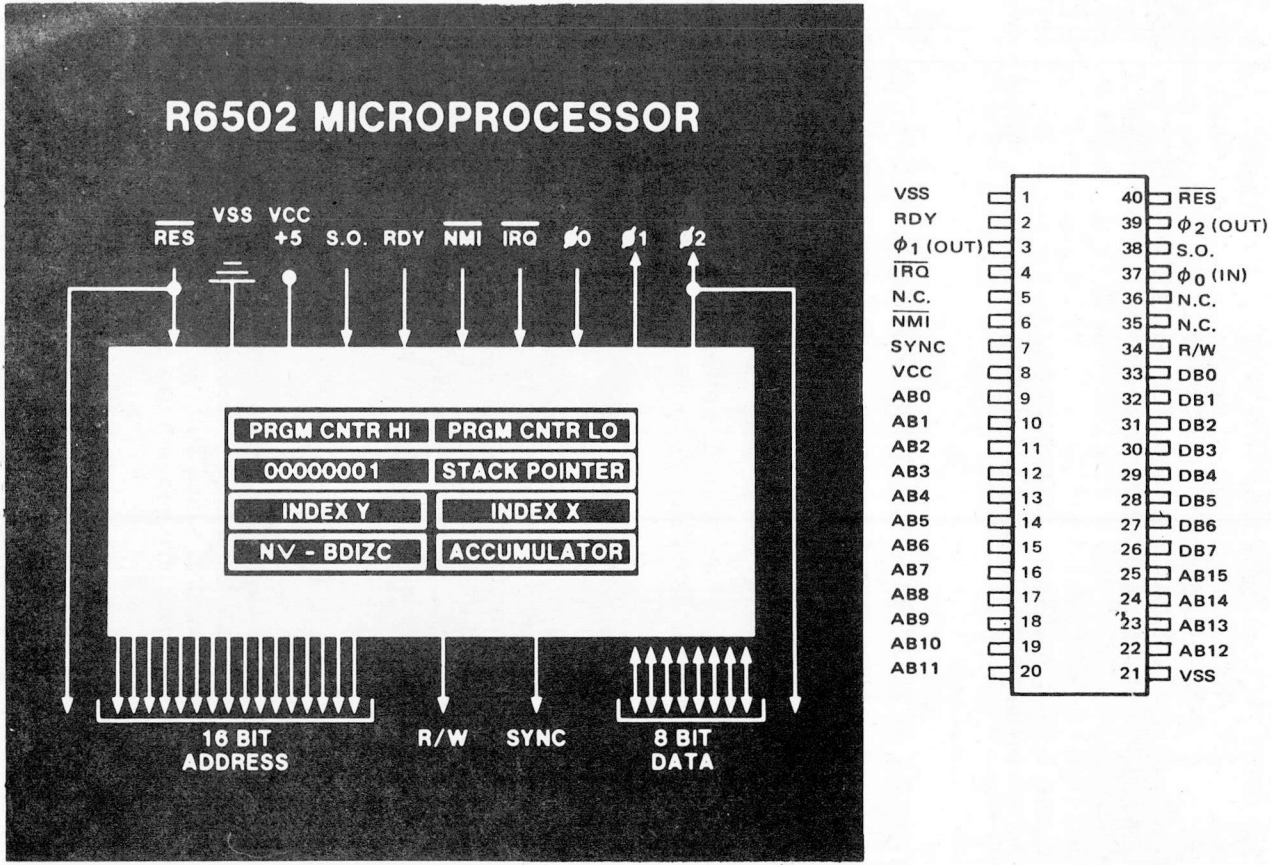


Fig. 5-7.–Estructura simplificada del R 6502, a la izquierda y configuración de los pines del chip, a la derecha.

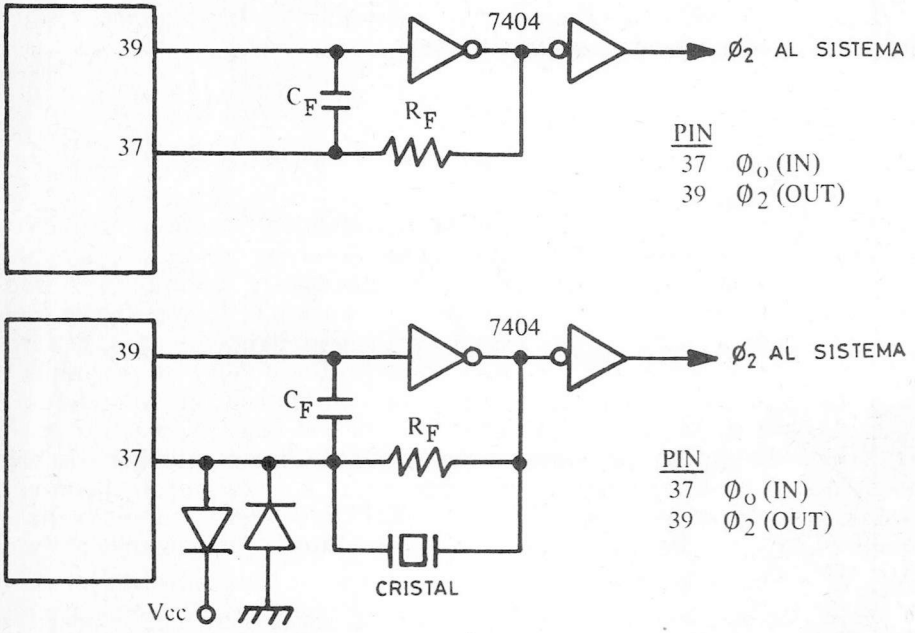


Fig. 5-8.–Esquemas del generador de la base de tiempo, ϕ_2, mediante un cristal externo, en la parte superior, o por medio de una red RC, en la parte inferior.

desde la CPU a los periféricos y a la memoria. En las salidas del bus hay buffers amplificadores del tipo tri-estado, capaces de soportar una carga estándar TTL y 130 pF. Los buffers tri-estado permanecen en "estado flotante", a excepción del tiempo en el que transmiten datos. A la frecuencia de 1 MHz los datos permanecen estables en el bus 100 ns antes de finalizar la fase ϕ_2.

RDY

Mediante la señal de este pin, el microprocesador puede adaptarse a la baja velocidad de las memorias, así como al Acceso Directo a Memoria, DMA. Esta señal de entrada permite detener al microprocesador en todos los ciclos, a excepción de los de escritura. El estado en que entra la CPU con la señal RDY permite a los buses de direccionamiento y de datos quedar libres, para poderse emplear desde el exterior.

Si la línea RDY pasa a bajo, que es el nivel activo, durante un ciclo en que se haya iniciado la operación de escritura, el procesador no se

detiene, ejecutará dicha operación y se detendrá en la próxima de lectura, identificada por ser la señal R/W = 1.

$\overline{IRQ}$. Orden de interrupción "mascarable"

Esta entrada de nivel TTL ordena el comienzo de una interrupción. Antes de ser reconocida y tratada la señal de interrupción, el microprocesador completa la instrucción normal, que se está ejecutando. A continuación, se examina el bit o flag de interrupción I, del registro de estado. Si dicho flag está a nivel 0, se autoriza la interrupción, que, por este motivo, recibe el nombre de "mascarable". El contador de programa y el registro de estado se salvan o almacenan en la posición que indique el puntero del stack, mientras que la CPU pone el flag I a nivel alto, para evitar que se produzcan nuevas interrupciones. Finalmente, la parte baja o de menos peso del contador de programa, PCL, se carga con el contenido de la posición FFFE y la parte alta, PCH, con el contenido de la dirección FFFF, transfiriendo, de este modo, el control del programa al "vector memoria" localizado en esas direcciones.

Para poderse detectar cualquier señal de interrupción, la señal RDY deberá estar en nivel alto.

$\overline{NMI}$. Interrupción "no mascarable"

Esta línea permite una interrupción de carácter incondicional, es decir, es similar a la $\overline{IRQ}$, pero su ejecución no depende del estado del flag I, pues se realiza siempre.

El contenido del contador del programa, PC, se carga con el vector dirección, definido por las posiciones FFFA y FFFB, para la parte de más y menos peso respectivamente.

En resumen, hay dos líneas externas de la CPU a implementar con hardware, que permiten dos tipos de interrupciones: $\overline{IRQ}$, interrupción mascarable, que depende del estado del flag I del registro de estado, y $\overline{NMI}$, interrupción no mascarable o incondicional. En ambos tipos de interrupción el contador de programa se carga con el contenido de dos posiciones especiales de la memoria, produciendo una especie de "salto" en el programa, que se está ejecutando, con el que se traslada a una subrutina de tratamiento de la interrupción.

R/W. Read /Write (Lectura-Escritura)

Esta línea permite al procesador controlar la dirección de transferencia de los datos entre la CPU y los restantes elementos. Permanece en nivel alto, excepto cuando se realiza una operación de escritura de datos, bien en la memoria o en los elementos periféricos.

S.O. Flag de sobrepasamiento

Esta señal de nivel TTL permite un control externo del bit de sobrepasamiento del registro de estado.

$\overline{\text{RES}}$. Reset

La línea $\overline{\text{RES}}$ se usa para inicializar al microprocesador.

Cuando la línea $\overline{\text{RES}}$ pasa al nivel activo (bajo), comienza la secuencia de reset, que consiste en cargar al contador de programa, PC, tras un retardo de 6 ciclos de máquina, con la posición de inicio del programa, la cual está almacenada en las posiciones de memoria FFFC y FFFD. El tra tratamiento de esta señal es semejante a las de interrupción: $\overline{\text{IRQ}}$ y $\overline{\text{NMI}}$.

SYNC

Esta línea de salida de la CPU se usa para identificar los ciclos, en los que el microprocesador realiza la búsqueda del código OP. La señal SYNC sube durante la fase ϕ_1 de búsqueda de un código OP y permanece en ese estado mientras dura dicho ciclo.

DBE. Habilitación del bus de datos

Permite desde el exterior habilitar el bus de datos. Para sistemas que precisan más tiempo, que el fijado por la señal de reloj, se puede acceder al bus de datos, habilitándolo mediante la línea DBE.

CICLOS DE LECTURA Y ESCRITURA. DIAGRAMAS DE TIEMPO

Las 16 líneas del bus de direcciones tienen un carácter unidireccional, puesto que su estado se gobierna únicamente desde la CPU, que seleccio-

na hasta 64 K posiciones de memoria. Por el contrario, el bus de datos, compuesto por 8 líneas, tiene carácter bidireccional, ya que sus líneas transfieren datos o instrucciones desde la memoria a la CPU en los ciclos de lectura, mientras que en los ciclos de escritura los datos se trasladan desde la CPU hasta la memoria. También existe transferencia bidireccional de datos con los módulos de entrada y salida.

La dirección de los datos se controla con la línea R/W. Cuando R/W está a nivel alto, lo que supone la existencia de un voltaje superior a 2,4 V, los datos de la memoria se trasladan al procesador (Ciclo de lectura). Si R/W está a nivel bajo, se procede a un Ciclo de escritura de los datos en la memoria.

En la mayoría de los sistemas microcomputadores la transferencia de datos se controla mediante señales de reloj. Las señales de reloj consisten en dos ondas cuadradas, aproximadamente iguales y opuestas. Las ondas aludidas se denominan ϕ_1 y ϕ_2 y se generan a base de una señal externa ϕ_0, que estabiliza la frecuencia.

En la figura 5-9 se muestra el diagrama de las señales de reloj ϕ_1 y ϕ_2, en el período de un ciclo (T_{CYC}).

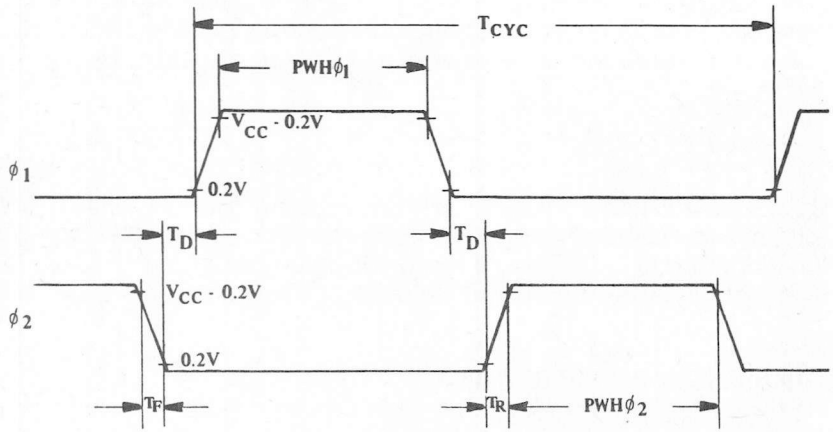


Fig. 5-9.–Diagrama de tiempos de las fases ϕ_1 y ϕ_2 durante el período de un ciclo.

En las figuras 5-10 y 5-11 se presentan los diagramas de tiempo del bus de direccionamiento y del bus de datos, así como el estado de las líneas R/W y DBE, durante un ciclo de lectura y durante un ciclo de escritura. Las especificaciones, que se ofrecen en las figuras, se refieren a un funcionamiento con una frecuencia de 1 MHz de la señal de reloj.

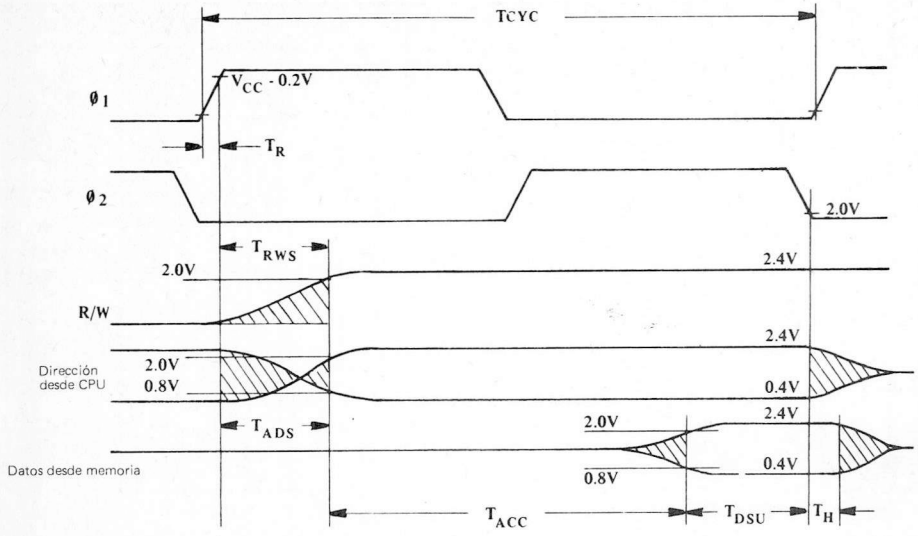


Fig. 5-10.–Diagrama de tiempos durante un ciclo de lectura.

El bus de direccionamiento tiene garantizado su estado estable 300 ns después del cambio de ϕ_1, mientras que el bus de datos lo tiene 10 ns antes de la transición de ϕ_2. Al trabajar con una frecuencia de 1 MHz, se permite que los elementos de memoria entreguen los datos al bus de datos en, aproximadamente, 575 ns.

Con respecto a las figuras anteriores, en la tabla que se muestra seguidamente se indican el significado de los símbolos utilizados, así como sus características más representativas con respecto a su duración.

CARACTERISTICA	SIMBOLO	MIN.	TIP	MAX.	UNIDAD
Tiempo del ciclo	T_{CYC}	1,0	—	—	μs
Anchura del impulso de reloj $\phi 1$ (Medido con: V_{CC} −0,2 V) $\phi 2$	PWH $\phi 1$ PWH $\phi 2$	430 430	—	—	ns
Intervalo subida y bajada (Medio desde 0,2 V a V_{CC} −0,2 V)	T_F, T_R	—	—	25	ns
Tiempo de retardo entre clocks (Medido con 0,2 V)	T_D	0	—	—	ns

CARACTERISTICAS	SIMBOLO	MIN.	TIP	MAX.	UNIDAD
Tiempo de preparación (Set up) de Lectura/Escritura	T_{RWS}	—	100	300	ns
Tiempo de preparación (Set up) de la Dirección	T_{ADS}	—	200	300	ns
Tiempo de acceso a la lectura de memoria T_R $T_{CYC} - (T_{ADS} - T_{DSU} - tr)$	T_{ACC}	—	—	575	ns
Período de tiempo de estabilidad de Datos	T_{DSU}	100	—	—	ns
Tiempo de sujección (Hold) de Datos	T_H	10	30	—	ns
Tiempo de Enable alta para entrada de DBE	T_{EH}	430	—	—	ns
Tiempo de preparación de Datos	T_{MDS}		150	200	ns

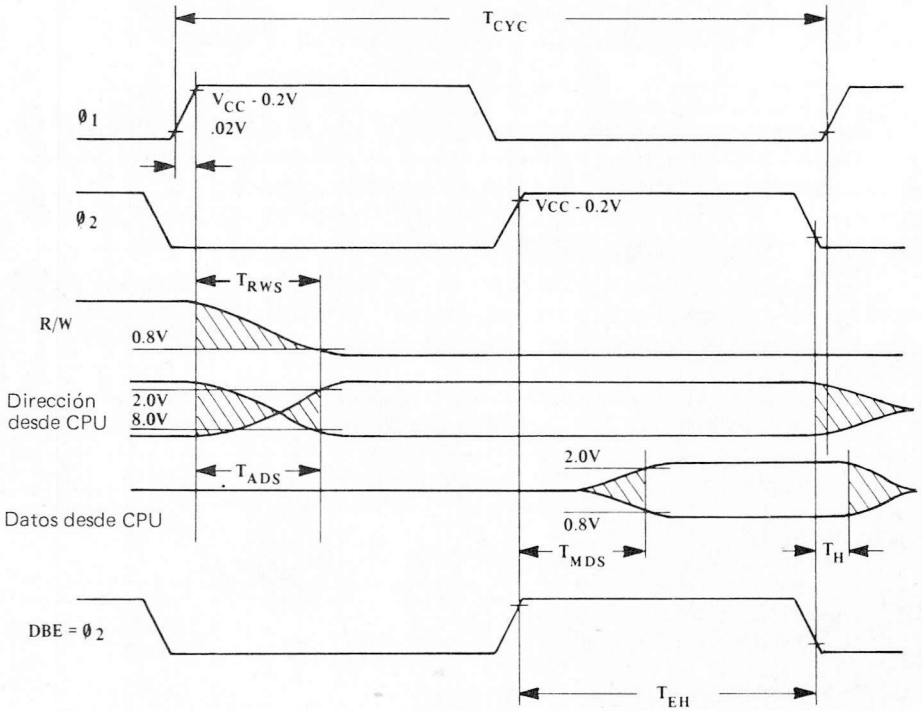


Fig. 5-11.–Diagrama de tiempos durante un ciclo de escritura.

SISTEMA CON MICROPROCESADOR

De lo expuesto hasta el momento se puede deducir, que un sistema completo a base de un microprocesador de la familia R 65000, como por ejemplo el R 6502, puede estructurarse tal como se ha esquematizado en la figura 5-12.

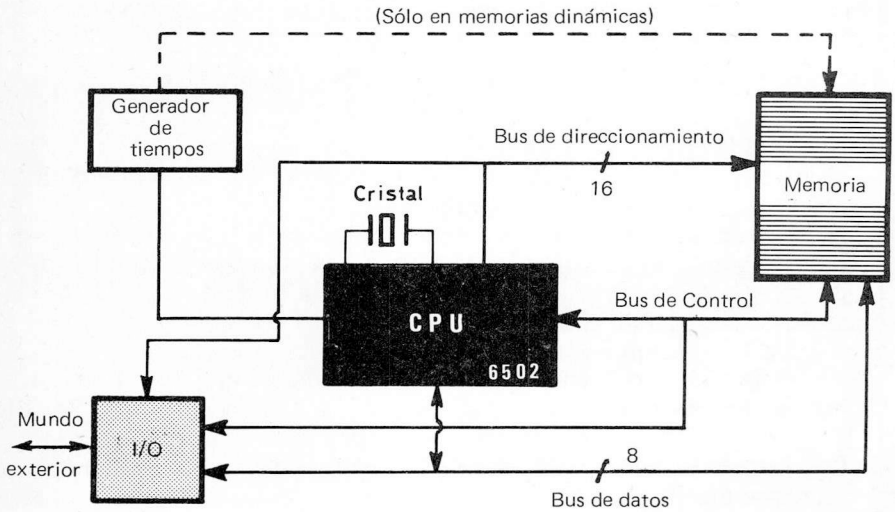


Fig. 5-12.–Sistema completo a base del microprocesador R 6502.

2.ª PARTE: MICROPROCESADOR MOTOROLA 6800

INTRODUCCION

El microprocesador 6800 fue fabricado en 1974 por Motorola. En la actualidad existen otras marcas, como AMI y Hitachi, que actúan como segundas fuentes. Se trata de un microprocesador muy popular en todo el mundo y es ampliamente empleado.

Las características generales más destacables del 6800 son:

- Alimentación única de +5 V.
- Niveles lógicos TTL de adaptación.
- Tiempo para la realización de las instrucciones, comprendido entre un mínimo de 2 μs y un máximo de 12 μs.
- Frecuencia de 1 MHz de la señal de reloj.
- Paralelamente al microprocesador 6800, existen otros circuitos integrados compatibles con él, que resuelven la mayor parte de los circuitos auxiliares de aplicación.
- La palabra de trabajo está formada por 8 bits.
- El Decodificador de instrucciones del 6800 acepta 72 instrucciones, que se convierten, en realidad, en 197, teniendo en cuenta los diversos modos de direccionado, que admite cada una.
- Capacidad de memoria de 64 K, al disponer de un bus para el direccionamiento de 16 líneas.
- Amplio stack.
- Tratamiento de las entradas y salidas de datos como posiciones de memoria. A este sistema se le denomina "mapa de memoria".
- Extensa información técnica.

ARQUITECTURA GENERAL DEL MICROPROCESADOR 6800

Toda CPU o microprocesador contiene una serie de elementos fundamentales, que responden al esquema básico mostrado en la figura 5-13.

Cada fabricante en cada uno de sus modelos de microprocesador enriquece la arquitectura básica, mostrada en la figura 5-13, dotándola de elementos específicos, con objeto de favorecer las funciones particulares, que mejoran el fin para el que se diseñó. Este es el caso del microprocesador MC-6800, que responde a una arquitectura fundamental, como es la mostrada en la figura 5-14.

Los conocimientos, hasta ahora adquiridos, son suficientes para comprender el funcionamiento de todos los componentes del MC-6800. No se

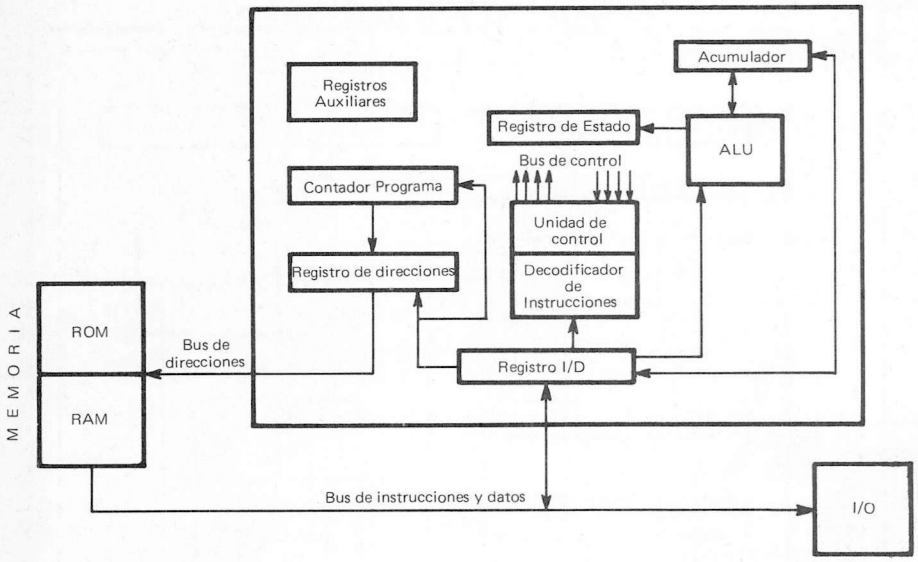


Fig. 5-13.–Constitución general básica de un microprocesador y su interconexión con los restantes componentes que forman un microcomputador.

comentan la misión y características del "registro de direcciones", del "registro de instrucciones y datos" del "Decodificador de instrucciones" y de la "Unidad de Control", por ser iguales en todos los microprocesadores. Seguidamente se exponen las particularidades de los restantes componentes del MC-6800.

Contador de Programa, PC: Sirve para direccionar la posición de la memoria de instrucciones, donde se encuentra el programa. Consiste en un registro de 16 bits, capaz de direccionar 2^{16} posiciones de memoria, equivalentes a 65.536 palabras de 8 bits (64 K).

Registro de direcciones: También está formado por 16 bits y en él se guarda el código de la dirección de memoria, que se va a enviar a través del bus de direcciones.

Acumuladores A y B: El microprocesador MC-6800 dispone de dos Acumuladores de 8 bits cada uno, que se denominan A y B. Las funciones de ambos acumuladores son similares y se gobiernan mediante las mismas instrucciones, que sólo se diferencian en sus códigos nemónicos por las letras A y B, que hacen referencia a cada uno.

La existencia de 2 acumuladores aumenta extraordinariamente la potencia de las operaciones lógico-aritméticas.

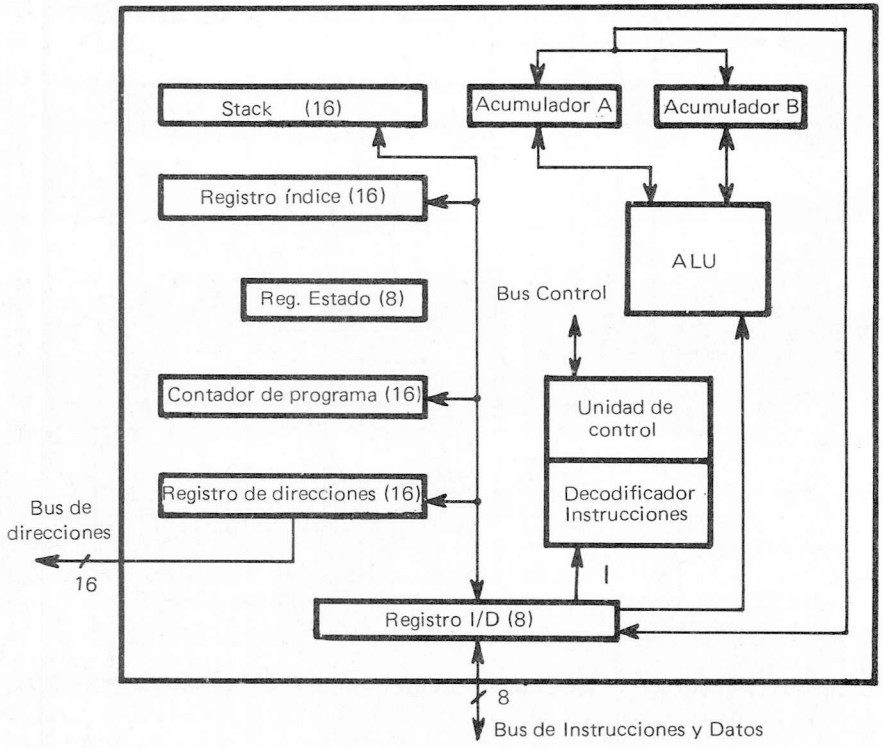


Fig. 5-14.–Arquitectura del microprocesador MC-6800.

Registro de Estado: Es un registro de 8 bits, 6 de los cuales se emplean como flags o indicadores del estado interno de la CPU. En la figura 5-15 se presenta la distribución y representación de los bits, que conforman este registro.

1	1	H	I	N	Z	V	C

Fig. 5-15.–Distribución y representación de los bits del registro de Estado.

En general, cada bit, que actúa como indicador en el registro de Estado, pasa a contener un nivel lógico 1, cuando se produce una condición al finalizar una instrucción. El cometido de cada uno de dichos bits es el siguiente:

C: Flag de acarreo o carry. Sirve para indicar el acarreo, si lo hay, del bit de más peso del Acumulador.
V: Flag de sobrepasamiento. Indica el desbordamiento de la capacidad del Acumulador.
Z: Flag cero. Este bit pasa a contener un nivel 1, cuando el Acumulador ha quedado, de la operación anterior, con todos sus bits en cero.
N: Flag de signo. Indica si el resultado es positivo o negativo (0 ó 1).
I: Flag para las interrupciones enmascarables. Es un flag cuyo estado permite, o no, la ejecución de las interrupciones enmascarables.
H: Flag de acarreo intermedio. Indica el acarreo, si lo hay, del 4.º bit del Acumulador.

Los dos restantes bits del registro de Estado no son representativos y contienen siempre el nivel 1.

La mayor parte de los flags pueden ponerse a 1 (set) o a 0 (clear), mediante instrucciones.

Registro Puntero de Stack: Se trata de un registro de 16 bits, que se emplea para direccionar cualquier zona de la memoria, como si se tratase de una memoria tipo LIFO (último en entrar, primero en salir). Su principal misión es la de salvar el contenido de los registros principales de la CPU, cuando se rompe la secuencia de un programa y, posteriormente, hay que regresar al estado en que había quedado el microprocesador, cuando se produjo el salto.

Registro Indice: Se trata de un registro, que posee 16 bits y tiene capacidad para incrementarse y decrementarse. Se emplea para llevar a cabo el direccionamiento de la memoria por el modo denominado "indexado", que se explicará más adelante.

Buffers de los buses de datos y de direcciones: La transferencia de datos, instrucciones y direcciones entre la CPU y los restantes elementos del sistema, se realiza a través de aisladores-amplificadores (buffers) triestado, que trabajan con niveles TTL. Dichos buffers no se han señalado en la figura 5-14, pero están situados a la salida de los buses en la CPU.

Se puede impedir la transferencia de datos entre la CPU y el bus de datos, pasando al buffer correspondiente al estado de alta impedancia, lo que se consigue poniendo a nivel bajo la línea de entrada a la CPU, llamada DBE, que se comentará seguidamente. De forma similar, el buffer de direcciones pasa al estado de alta impedancia, independizando al bus de la CPU, cuando se aplica un nivel bajo a la línea de entrada a la CPU, llamada TSC.

DIAGRAMA DE CONEXIONADO Y FUNCION DE LOS PINES DEL MC-6800

En la figura 5-16 se muestra el diagrama del conexionado de los pines, que constituyen la configuración externa del circuito integrado MC-6800.

De los 40 pines o patillas externas del circuito integrado, que contiene al microprocesador MC-6800, 2 de ellas no se emplean, 8 corresponden al bus de datos, 3 sirven para recibir la alimentación, 16 pertenecen al bus de direccionamiento de la memoria y el resto de los pines responden al siguiente cometido:

HALT: Cuando esta línea de entrada se activa (nivel bajo, al representarse con la barra superior que indica complementación), la CPU entra en

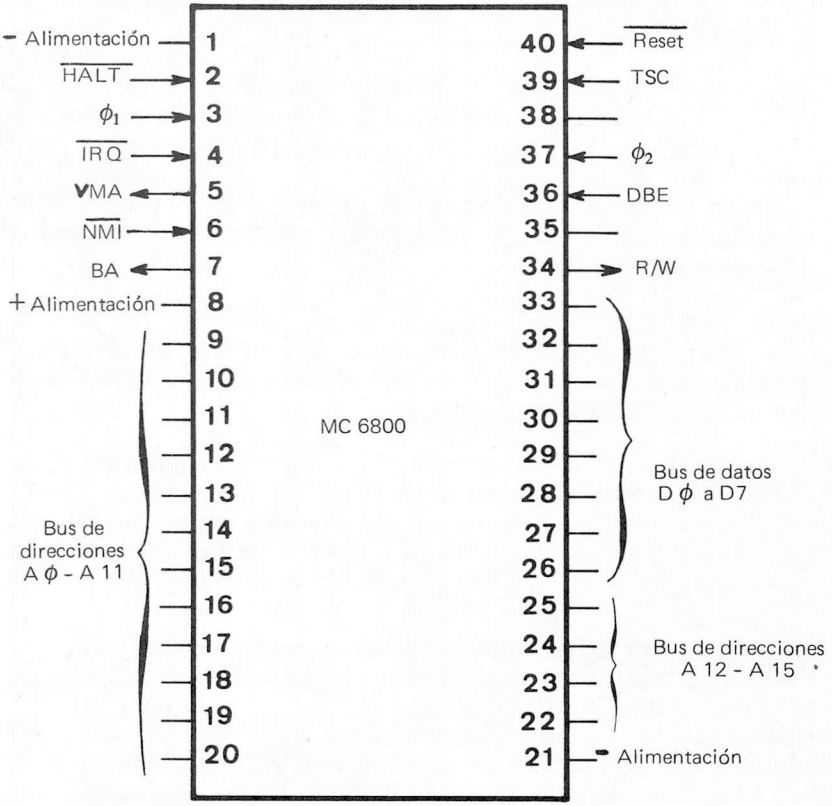


Fig. 5-16.–Designación de los pines del MC-6800.

un estado de parada, llamado HALT. Es una forma de detener a la CPU y desconectarla de los buses de direccionamiento y de datos, al quedar éstos en el estado de alta impedancia.

ϕ_1 *y* ϕ_2: Son las señales de reloj, que determinan la duración de los ciclos de la CPU y sirven para sincronizar al sistema. La frecuencia normal del reloj es de 1 MHz y se consigue a través de un circuito externo.

Hay 3 tipos de ciclos que conforman a las instrucciones: el de lectura, el de escritura y el interno. En general la fase ϕ_1 coincide con el ciclo de búsqueda de una instrucción y el ϕ_2 con la de ejecución.

Los estados de las diversas líneas, que intervienen en un ciclo de lectura, se representan en la figura 5-17.

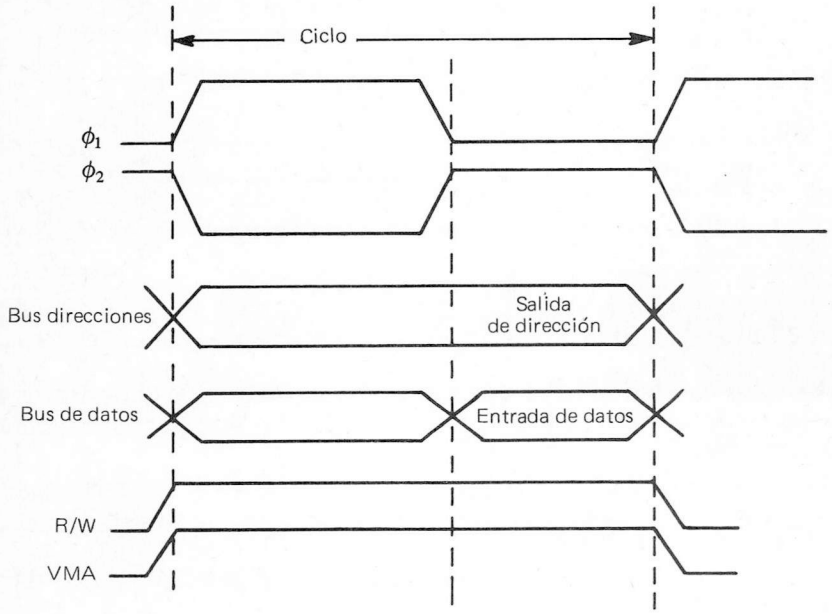


Fig. 5-17.–Diagrama de tiempos de un ciclo de lectura.

El significado de las líneas R/W y VMA se explicarán a continuación.

En la figura 5-18 se presentan, de forma similar a la figura anterior, los estados de las líneas más representativas de la CPU durante un ciclo de escritura. El dato debe permanecer estable sobre el bus de datos como mínimo, el tiempo que necesita la memoria para tomar y grabar el dato.

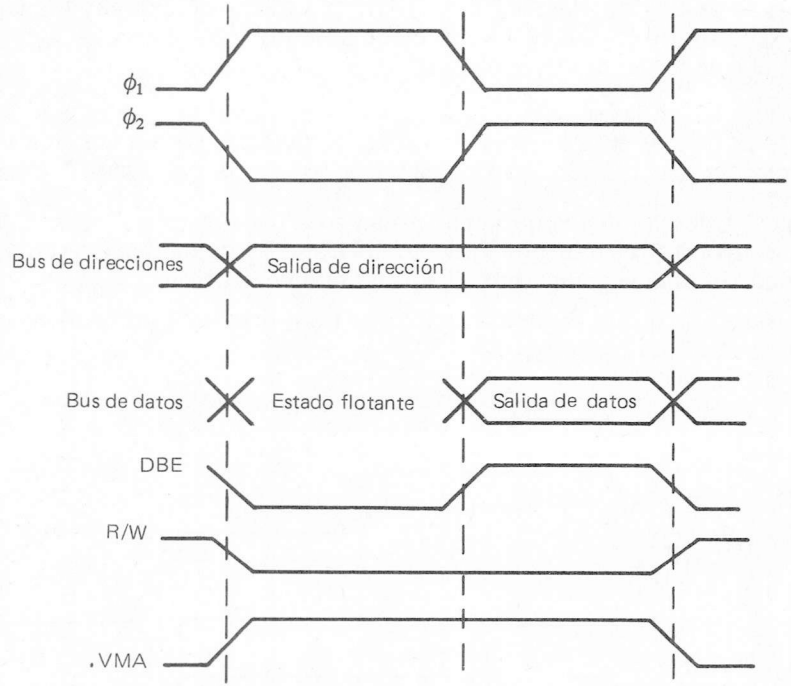


Fig. 5-18.–Diagrama de tiempos de un ciclo de escritura.

R/W: En el estado activo (nivel alto) indica lectura de la memoria y en el nivel bajo, escritura. El nivel de alta impedancia se puede dar, cuando se detiene la CPU.

TSC: Cuando esta línea se pone en nivel 1, pasan al estado de alta impedancia el bus de direccionamiento y la línea R/W, mientras que las líneas VMA y BA se fuerzan al nivel 0. La línea TSC se usa para hacer posible el acceso directo a la memoria o DMA y se denomina "Control triestado".

$\overline{RESET}$: Esta línea se emplea para inicializar el sistema con microprocesador. Cuando se activa (nivel bajo), la CPU inicia la secuencia de RESET, cargando al PC con el contenido del vector de memoria, formado por las posiciones FFFE y FFFF, en las que está almacenada la dirección de la 1.ª instrucción, asimismo, pasan a cero los registros, contadores y flags.

La operación de RESET también se realiza automáticamente, al conectar la alimentación al microprocesador.

VMA (Dirección válida de memoria): La activación de esta línea sirve para indicar en el exterior de la CPU la validez de la dirección, que se transfiere por el bus de direcciones a los elementos periféricos y memoria.

BA (Bus disponible): Cuando esta línea pasa a nivel 1, significa que el microprocesador está detenido, por ejemplo, en el estado HALT. Los buses quedan desconectados al parar la CPU y pueden emplearse para otros propósitos, como puede serlo el DMA. Luego la línea BA avisa al exterior, cuando se detiene la CPU.

DBE: Activa o desactiva desde el exterior al bus de datos. Se emplea para pasar al estado de alta impedancia al bus de datos y aprovecharlo, por ejemplo, para realizar el DMA, u otra operación desde el exterior.

$\overline{NMI}$: Cuando se activa esta línea con un nivel bajo, se efectúa una secuencia de "interrupción no enmascarable", lo que significa que siempre se produce y no depende del estado del flag I del registro de Estado.

Esta interrupción produce la carga del PC con el contenido del vector memoria, formado por las direcciones FFFC y FFFD, que guardan la dirección de la 1.ª posición de la rutina de tratamiento para esta interrupción.

$\overline{IRQ}$: Cuando desde el exterior se activa esta línea con un nivel bajo, se efectúa una secuencia de "interrupción enmascarable", lo que significa que sólo puede llevarse a cabo, si el flag I del registro de Estado está a nivel 0. El vector memoria para realizar el tratamiento de este tipo de interrupción está localizado en las posiciones FFF8 y FFF9.

Al comenzar una interrupción, enmascarable o no, se salvan mediante el puntero del stack los contenidos existentes en ese momento en el PC, en el registro Indice, en el registro de Estado y en los dos acumuladores.

Como colofón de la arquitectura del 6800, nos referiremos brevemente a sus posibles interrupciones, que son las siguientes:

a) Interrupción no enmascarable. Producida desde el exterior a través de la línea $\overline{NMI}$.

b) Interrupción enmascarable. Producida desde el exterior a través de la línea $\overline{IRQ}$.

c) Interrupción por programa, mediante la instrucción SWI.

d) RESET, que procede a la inicialización del sistema, que actúa por la línea $\overline{RESET}$ y también cuando se conecta la alimentación.

Cada vez que la CPU reconoce y acepta una de las interrupciones expuestas, realiza las siguientes operaciones:

1.ª) Se aborta la instrucción que había ejecutado la fase de búsqueda, pero completando la ejecución de la instrucción en curso.

2.ª) Utilizando el punto del stack, se salvan los contenidos del PC, el registro Indice, el de Estado y el de los dos acumuladores.

3.ª) Se colocan en el PC (16 bits), el contenido de las dos posiciones de memoria (vector memoria), asignadas para cada interrupción y que se citan a continuación:

$\overline{IRQ}$: FFF8 y FFF9
$\overline{NMI}$: FFFC y FFFD
$\overline{RESET}$: FFFE y FFFF
Instrucción SWI: FFFA y FFFB

4.ª) Comenzando en la dirección señalada por el vector memoria, se realiza la subrutina de tratamiento de la interrupción. Cuando la interrupción la ha podido originar cualquiera de los varios periféricos, la subrutina comenzará detectando al elemento causante de la interrupción, para tratarla adecuadamente.

3.ª PARTE: MICROPROCESADOR 8080 DE INTEL

ARQUITECTURA DEL 8080

Otro microprocesador conocido mundialmente es el 8080, de la casa INTEL, pionera en la comercialización de micros. Una versión mejorada y más actual del 8080, es el 8085, entre los que existe una gran similitud.

La arquitectura interna de la CPU 8080, válida también para el 8085, es la que se muestra en la figura 5-19.

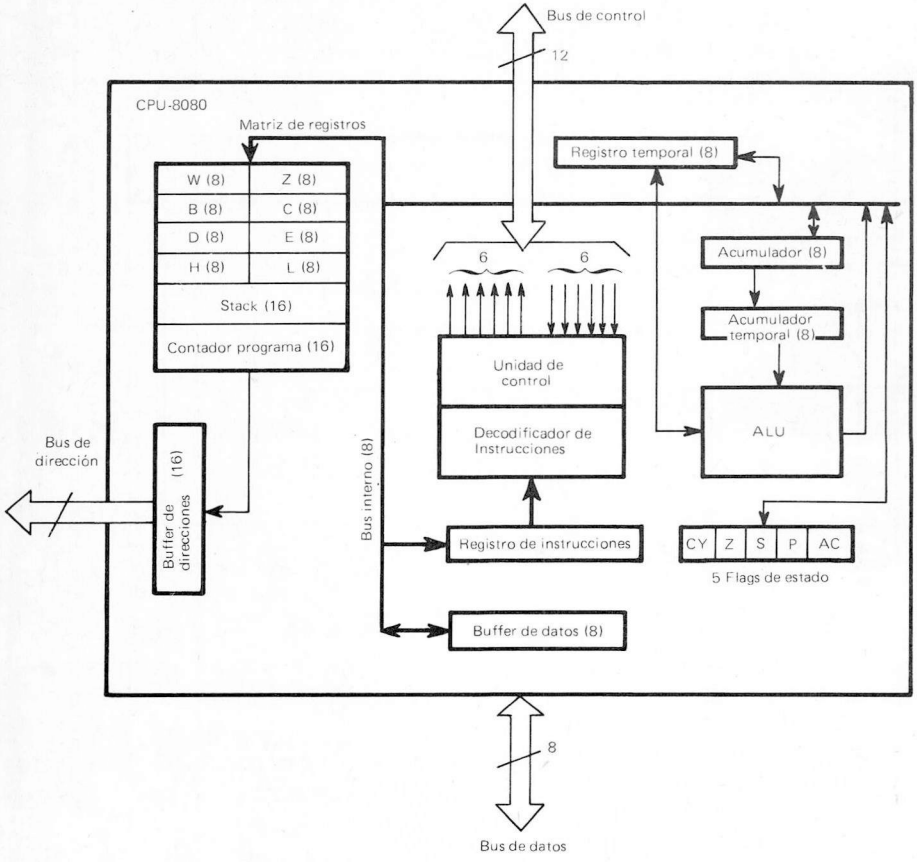


Fig. 5-19.–Arquitectura interna del microprocesador 8080 de Intel.

Las funciones correspondientes a la Unidad de Control, Decodificador de instrucciones y Registro de instrucciones, son semejantes al resto de los microprocesadores. Sin embargo, en el 8080 existen algunas peculiaridades en la ALU y en los diversos Registros de trabajo, que pasamos a comentar.

La sección de la ALU consta de:

1.°) ALU propiamente dicha, para el tratamiento de palabras de 8 bits en paralelo.
2.°) Acumulador principal de 8 bits, denominado A.
3.°) Acumulador temporal de 8 bits, ACT.
4.°) Registro temporal de 8 bits, TMP.
5.°) 5 registros de 1 bit, que configuran el denominado Registro de Estado y que actúan como flags o indicadores. La figura 5-20 presenta dicho Registro de Estado y la misión de cada uno de sus bits.

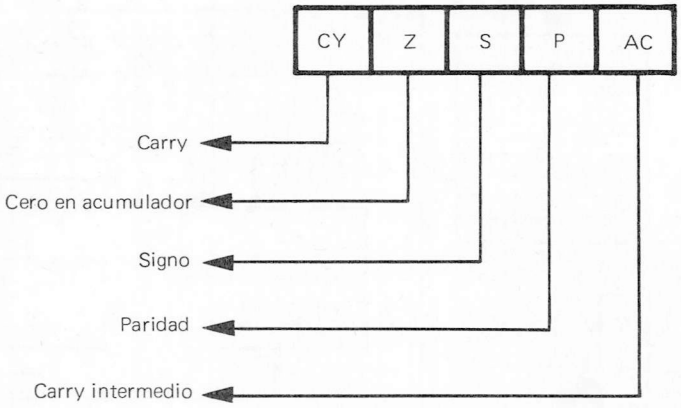


Fig. 5-20.–Significado de los flags que configuran el Registro de Estado.

El registro temporal, TMP, tiene por misión intercambiar su contenido con el bus interno, la ALU y el Registro de Estado.

Los dos datos, que acceden a la ALU para su tratamiento, lo hacen a través del Acumulador Temporal y el Registro Temporal. Los resultados se pueden depositar, como es normal, en el Acumulador o pasan directamente al bus interno de la CPU.

En la sección de registros del 8080, además del PC y del Stack, cuyas misiones son ya sobradamente conocidas, existen otros 4 registros más,

también de 16 bits, divididos en dos porciones de 8 bits cada uno. La organización de los registros B, C, D, E, H y L está diseñada para su actuación por parejas, mientras que W y Z actúan como registros temporales.

Los registros B, C, D, E, H y L se emplean fundamentalmente como elementos de apoyo en el direccionamiento de la memoria. Si actúan como registros individuales, transfieren su contenido de 8 bits al bus interno, pero, si actúan por parejas, los 16 bits pasan directamente hasta el buffer de dirección y, luego, al bus que controla.

Todos los registros tienen como característica común la de poder ser incrementados y decrementados fácilmente.

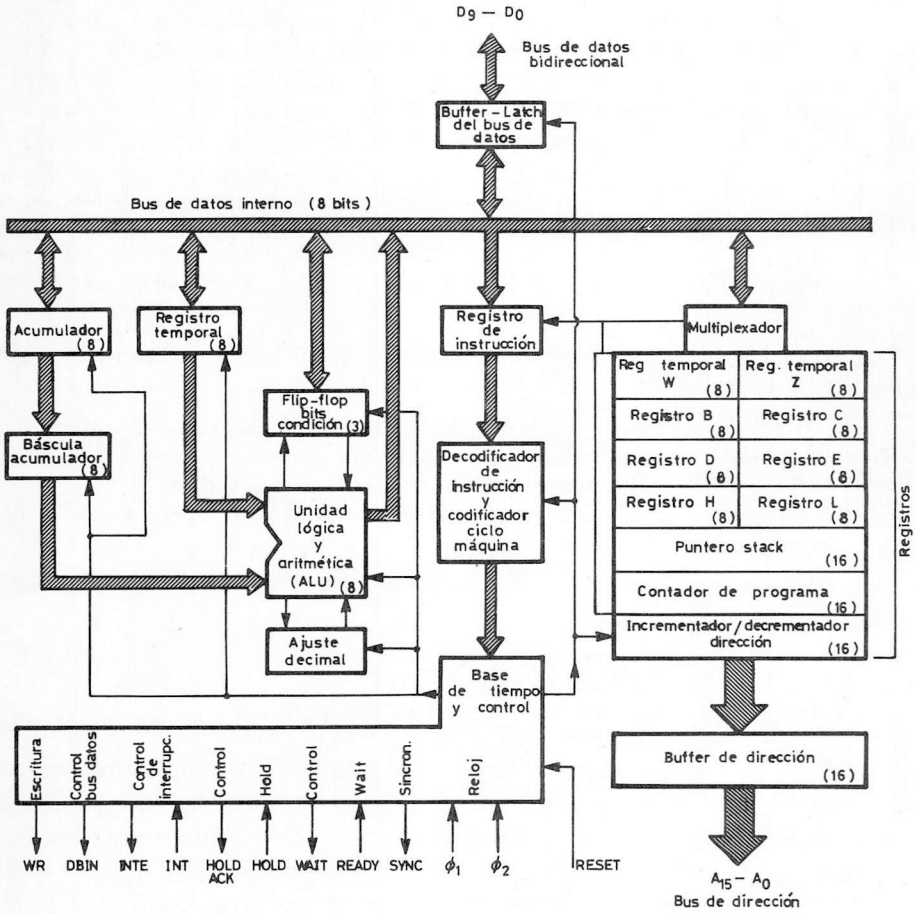


Fig. 5-21.–Arquitectura detallada del 8080.

En la figura 5-21 se muestra una arquitectura más completa del 8080 y de los diferentes buses y líneas, que le relacionan con los restantes elementos, que constituyen el sistema operativo.

El buffer del bus de datos de la figura 5-21 es de 8 bits, bidireccional y triestado, siendo utilizado como conexión entre el bus interno de la CPU y el bus de datos externo, formado por las líneas D0-D7.

Durante las operaciones de salida de la CPU, el contenido del bus interno de la misma se transfiere, a través de un latch de 8 bits, a los buffers de salida, que permanecen inhibidos durante las operaciones de no transferencia y en las de entrada de datos a la CPU.

En las operaciones de entrada a la CPU, la transferencia de datos tiene lugar desde el bus de datos externo al bus de datos interno de la CPU. Este se precarga al principio de cada estado interno, excepto en el estado T_3, que luego se explicará.

DIAGRAMA DE CONEXIONADO Y MISION DE LAS PATILLAS DEL 8080

La configuración externa, o diagrama de pines del 8080 se presenta en la figura 5-22.

La asignación de los 40 pines del 8080 es la siguiente, tal como se deduce de la figura 5-22:

a) 16 pines para el bus de direccionamiento de la memoria.
b) 8 pines para el bus de datos bidireccional.
c) 4 pines para la alimentación: +12 V, +5 V, –5V y masa.
d) 12 pines correspondientes al bus de control: 6 de entrada y 6 de salida.

Las 6 líneas de entrada del bus de control son:

1) Señal de reloj ϕ_1.
2) Señal de reloj ϕ_2.
3) RESET: Para inicializar la CPU.
4) INT: Línea que sirve para efectuar interrupciones.
5) HOLD: Línea para la petición del estado HOLD.
6) READY: Línea que sirve para sincronizar la CPU con las memorias lentas.

El cometido de las 6 líneas de salida del bus de control es el siguiente:

1) $\overline{WR}$: Indica si se procede a un ciclo de escritura.
2) SYNC: Sincroniza a la CPU con el resto de los componentes del sistema microcomputador, o sea, memorias, módulos I/O, etc.

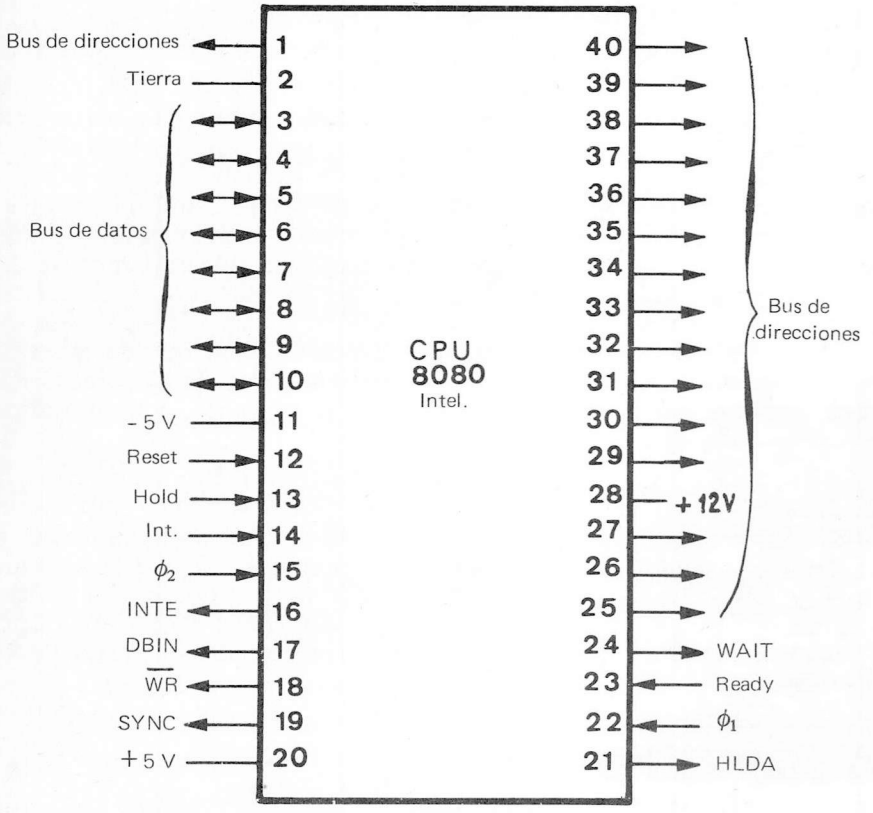


Fig. 5-22.–Configuración externa del circuito integrado 8080.

3) INTE: Indica cómo se encuentra el estado de las interrupciones.
4) HOLDA: Su nivel indica si la CPU se halla o no en el estado HOLD.
5) WAIT: Sirve para indicar si la CPU se encuentra en el estado WAIT.
6) DBIN: Indica que el sentido del movimiento de la información en el bus de datos es de entrada.

CICLOS DEL PROCESADOR 8080

Un ciclo de instrucción es el tiempo necesario para la búsqueda y ejecución de una instrucción.

Durante la operación de búsqueda o identificación de una instrucción determinada (de uno, dos o tres bytes), ésta se extrae de la memoria y se

coloca en el registro de instrucción de la CPU. Durante la ejecución, se decodifica la instrucción y se convierte en las operaciones específicas, que la misma determina.

Todas las instrucciones constan de uno, dos, tres, cuatro o cinco ciclos de máquina, teniendo en cuenta que, por ejemplo, en una operación de búsqueda de una instrucción se precisa de un ciclo de máquina por cada uno de los bytes, que se extrae de la memoria, y que, asimismo, se invierte un ciclo de máquina cada vez que la CPU accede a la memoria o a una puerta de entrada/salida. De lo expuesto se deduce, que el tiempo necesario en la ejecución de una instrucción es directamente proporcional a las veces que se accede a la memoria en la operación de búsqueda. Algunas instrucciones no requieren de ningún ciclo de máquina más del invertido en esta operación, en tanto que otras precisan ciclos de máquina adicionales, para escribir o leer datos en o desde la memoria, o algún dispositivo de entrada/salida.

Se denomina estado a la unidad menor de actividad de un proceso, definida como el intervalo entre dos flancos positivos de los pulsos de reloj ϕ_1. Estos pulsos, junto a los denominados ϕ_2, constituyen las dos fases de un oscilador, cuya frecuencia determina el reloj de la máquina. Combinando externamente estos pulsos, tal como se observa en la figura 5-23, se obtiene la nueva señal SYNC, que determina el comienzo de cada ciclo de máquina. Cada uno de éstos puede constar de tres, cuatro o cinco estados.

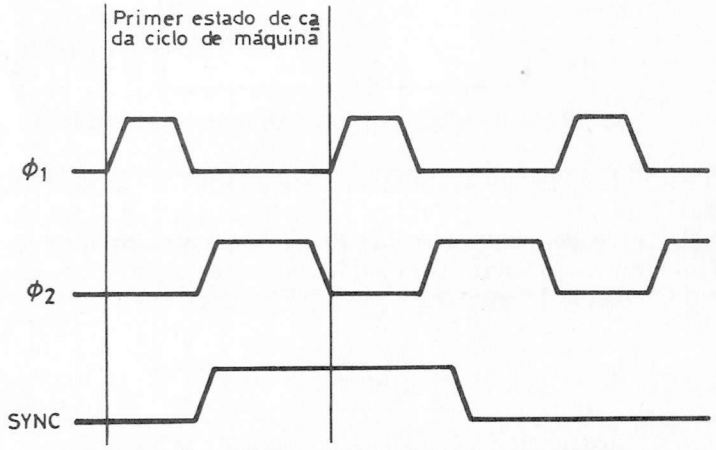


Fig. 5-23.—La señal SYNC determina el comienzo de cada ciclo máquina.

En esta forma se define la duración de un estado, y, por tanto, de un ciclo de máquina, así como la de ejecución de una instrucción. Existen tres excepciones, no obstante, en la duración de un estado. Estas son: el estado de espera (WAIT), el estado de paro (HLDA) y el estado de alto (HLTA), cuyas funciones se definirán más adelante, y cuya duración es indefinida, puesto que están controladas exteriormente. Sin embargo, y a pesar de la indeterminación de la duración de los mismos, el hecho de estar sincronizados hace que ésta sea siempre un múltiplo entero de período de reloj.

En síntesis, se puede afirmar, que cada período de reloj constituye un estado; de tres a cinco estados dan lugar a un ciclo de máquina, y de uno a cinco de estos ciclos comprenden un ciclo de instrucción. Un ciclo de instrucción, por tanto, precisa desde cuatro a dieciocho estados para su realización, según el tipo de operación a ejecutar.

IDENTIFICACION DEL CICLO DE MAQUINA

Excepto en la instruccion DAD, que requiere dos ciclos de máquina adicionales en su ejecución, hay un dato, que determina cuántos ciclos de máquinas se necesitan en un ciclo de instrucción: el número de veces que el procesador debe hacer referencia a una dirección de memoria o a un dispositivo periférico direccionable.

Al igual que otros procesadores, la constitución del 8080 no permite enviar más que una sola dirección por ciclo de máquina, por lo que, si la identificación y ejecución de una instrucción precisan dos referencias a memoria, el ciclo de ejecución asociado a esta instrucción consta de dos ciclos de máquina. Si una instrucción precisa de cinco referencias, el ciclo de instrucción contendrá cinco ciclos de máquina.

Todo ciclo de instrucción realiza, como mínimo, una referencia a memoria, mientras se busca e identifica la instrucción a ejecutar. Este ciclo de máquina, por tanto, es el primero de cualquier instrucción, sea ésta del tipo que sea.

En ocasiones este ciclo de máquina es el único de que consta el ciclo de instrucción, como ocurre con la instrucción de suma simple de registros (ADD r). Esta es una instrucción de un solo byte, en la que el contenido de uno de los seis registros de trabajo de la CPU se suma al contenido del acumulador, guardándose el resultado de este último. Como toda la información necesaria para la ejecución de esta instrucción puede estar contenida en los ocho bits de que consta un byte de código de instrucción, únicamente necesita realizarse una referencia a memoria. El ciclo de instrucción consta, por tanto, de un solo ciclo de máquina, que a su

vez consta de cuatro estados: tres para extraer la instrucción de la memoria, más un estado adicional para realizar la suma.

Supongamos ahora que la suma debe realizarse entre el contenido del acumulador y el de una posición de memoria determinada (ADD M). El principio de funcionamiento es el mismo que en el ejemplo anterior, pero en este caso se precisa de otro ciclo de máquina, en el cual se da la dirección de memoria, donde está contenido uno de los sumandos.

La secuencia en la ejecución de esta instrucción es la siguiente: en primer lugar, el procesador accede a la memoria, enviando la dirección contenida en el contador de programa, a fin de tomar el byte correspondiente al código de la instrucción. Es el ciclo de búsqueda e identificación. Una vez que los ocho bits de código están en la CPU y la instrucción ha sido identificada, operación que consta de tres estados, como en el caso anterior, el procesador envía una nueva dirección de memoria contenida en los registros H y L. Los ocho bits de datos, obtenidos durante esta operación de LECTURA DE MEMORIA, se colocan en un registro temporal de la CPU, habiéndose utilizado en esta operación tres nuevos estados (períodos de reloj). En el séptimo y último estado de la instrucción se realiza la suma entre el contenido del registro temporal y el acumulador. Se puede deducir, por tanto, que esta instrucción precisa de dos ciclos de máquina (dos accesos a memoria) y siete estados para su ejecución completa.

Otra instrucción característica es la de "salvar" el contenido de los registros H y L, cuya ejecución requiere cinco ciclos de máquina. Durante el ciclo de ejecución de esta instrucción, el contenido de los registros H y L se deposita en dos posiciones de memoria consecutivas, la dirección de las cuales viene determinada por los bytes de dirección, almacenados en las dos posiciones siguientes a la que ocupa en la memoria el byte de código de operación de la instrucción. La secuencia de operaciones, que tiene lugar en la ejecución de esta instrucción, es la siguiente:

1) Un ciclo de búsqueda, que consta de cuatro estados. Durante los tres primeros, el procesador busca e identifica la instrucción a través del contenido del contador de programa, el cual se incrementa en una unidad. En el cuarto estado tiene lugar la decodificación interna de la instrucción.

2) Un ciclo de LECTURA DE MEMORIA, que consta de tres estados. Durante el mismo, el byte definido por el contador de programa (segundo byte de la instrucción), se extrae de la memoria y se deposita en el registro Z de la CPU. El contador de programa se incrementa de nuevo en una unidad.

3) Un nuevo ciclo de LECTURA DE MEMORIA, consistente, asimismo, en tres estados, en los cuales el byte definido por el contador de programa (tercer byte de la instrucción) se extrae de la memoria y se carga en el registro W de la CPU. El contador de programa se incrementa en una unidad, conteniendo ahora la dirección del primer byte de la siguiente instrucción.

4) Un ciclo de ESCRITURA DE MEMORIA, de tres estados, en el que el contenido del registro L se transfiere a la posición de memoria, definida por los registros W y Z. El último estado de este ciclo se utiliza para incrementar el par de registros W y Z para indicar la próxima dirección de memoria, en la que deben cargarse datos.

5) Un nuevo ciclo de ESCRITURA DE MEMORIA, que consta asimismo de tres estados, y en el que el contenido del registro H se transfiere a la nueva dirección de memoria, definida por el par de registros W y Z.

En resumen, la instrucción "SHLD" contiene cinco ciclos de máquina y necesita 16 estados para su ejecución.

El resto de instrucciones del 8080 están comprendidas entre los extremos tipificados por las instrucciones "ADD r" y "SHLD". Por ejemplo, las instrucciones de entrada (IN) y salida (OUT) necesitan tres ciclos de máquina: uno de búsqueda de la instrucción, uno de lectura de memoria para obtener la dirección del periférico a que se accede, y uno de entrada o salida, a fin de completar la transferencia de datos.

Teniendo en cuenta que ningún ciclo de instrucción está constituido por más de cinco ciclos de máquina, en la ejecución de una instrucción pueden existir los siguientes diez tipos de ciclos:

1) Búsqueda.
2) Lectura de memoria.
3) Escritura de memoria.
4) Lectura de stack.
5) Escritura en stack.
6) Entrada.
7) Salida.

8) Interrupción.
9) Alto.
10) Alto-interrupción.

El procesador identifica el ciclo de máquina que se está ejecutando, transmitiendo una palabra de ocho bits durante el primer estado de cada ciclo a través de las líneas D0-D7. Esta información se carga en básculas, utilizándose posteriormente en la generación de las señales adecuadas, por la lógica exterior. El estado de estos bits, en función del ciclo que se está ejecutando, se puede observar en la tabla 5-1 y figura 5-24.

La forma en que estas señales determinan el ciclo correspondiente es muy simple. Así, por ejemplo, la presencia del bit STACK (D2) determina indefectiblemente un ciclo de escritura en el stack. Si este bit, además, va acompañado del bit $\overline{WO}$ (D1) y MEMR (D7), el ciclo determinado es el de lectura de stack, y así sucesivamente hasta los diez ciclos de máquina posibles.

BIT DEL BUS DE DATOS	INFORMACION ESTADO	❶ BUSQUEDA INSTRUCCION	❷ LECTURA MEMORIA	❸ ESCRITURA MEMORIA	❹ LECTURA STACK	❺ ESCRITURA STACK	❻ LECTURA ENTRADA	❼ ESCRITURA SALIDA	❽ ACEPTACION INTERRUPCION	❾ ACEPTACION ALTO	❿ ACEPTACION INTERRUPCION EN ALTO
D_0	INTA	0	0	0	0	0	0	0	1	0	1
D_1	WO	1	1	0	1	0	1	0	1	1	1
D_2	STACK	0	0	0	1	1	0	0	0	0	0
D_3	HLTA	0	0	0	0	0	0	0	0	1	1
D_4	OUT	0	0	0	0	0	0	1	0	0	0
D_5	M_1	1	0	0	0	0	0	0	1	0	1
D_6	INP	0	0	0	0	0	1	0	0	0	0
D_7	MEMR.	1	1	0	1	0	0	0	0	1	0

(N) ← Palabra de estado

Tabla 5-1.–Formación Palabras Estado 8080.

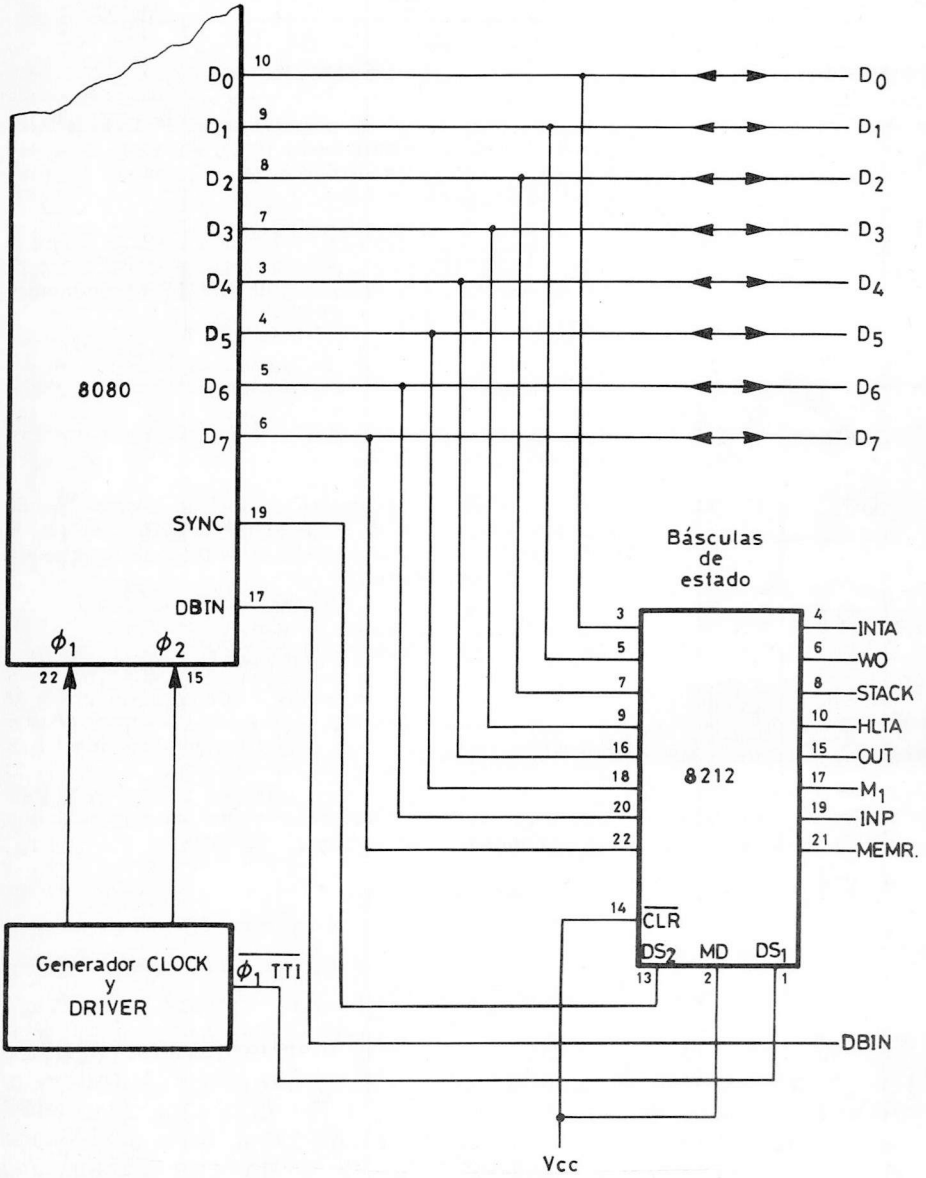


Fig. 5-24.–Generación Bits de Estado 8080.

DEFINICION DE LA INFORMACION DE ESTADO

SIMBOLO	BIT DEL BUS DATOS	DEFINICION
INTA*	D0	Señal de aceptación de una demanda de INTERRUPCION. Esta señal se utiliza para poner una instrucción de REINICIO (RST) en el bus de datos, cuando la línea DBIN está actuada.
$\overline{WO}$	D1	Indica que la operación en curso en el ciclo de máquina es de ESCRITURA de memoria o de SALIDA. Si no es así, se ejecutará una operación de LECTURA de memoria o de ENTRADA.
STACK	D2	Indica que en el bus de dirección hay la dirección del stack, obtenida en el Puntero de Stack.
HLTA	D3	Reconocimiento de la señal correspondiente a una instrucción de Alto (HLT).
OUT	D4	Indica que el bus de dirección contiene la dirección de un dispositivo de salida, y que el bus de datos contiene la información, que debe enviarse a ese dispositivo, cuando la línea $\overline{WR}$ está activada.
M1	D5	Indica que la CPU está en el ciclo de búsqueda (FETCH) del primer byte de una instrucción.
INP*	D6	Indica que el bus de dirección contiene la dirección de un dispositivo de entrada, y que los datos procedentes del mismo se colocarán en el bus de datos, cuando DBIN esté activado.
MEMR*	D7	Indica que el bus de datos contendrá datos procedentes de una operación de lectura de memoria.

* Estos tres bits de estado pueden usarse como control del flujo de datos en el bus de datos.

PROCESO DE TRANSICION DE ESTADOS

Cada uno de los ciclos de máquina de un ciclo de instrucción consta de tres a cinco estados, tal como se ha definido anteriormente. A estos estados se les define como T1, T2, T3, T4, T5 o TW. El número de estados, como se sabe, depende del tipo de instrucción. En la figura 5-25 puede observarse el diagrama de transición de estado a estado en el transcurso de un ciclo de máquina. Asimismo se observa cómo reaccionan las líneas READY, HOLD e INTERRUPT, que se discutirán más adelante.

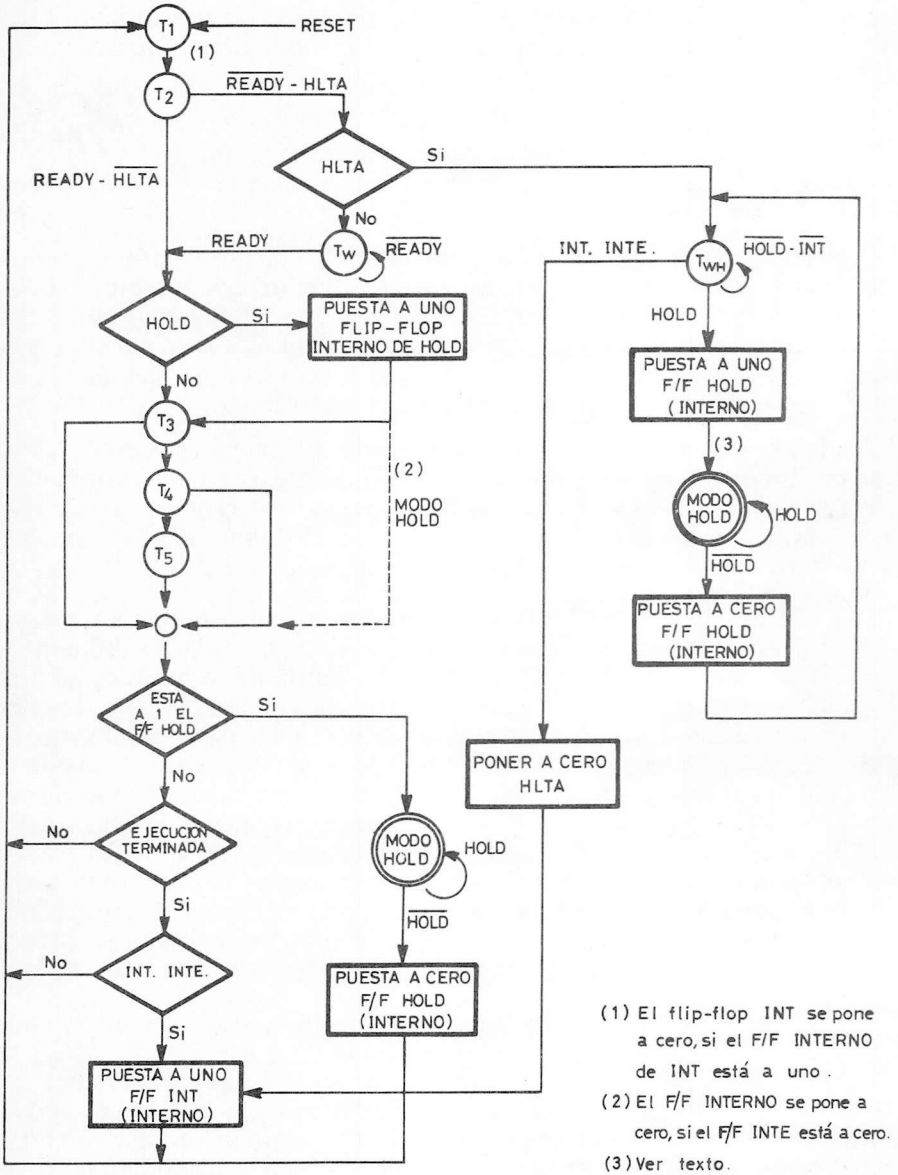


Fig. 5-25.–Diagrama Transición Estado.

La Unidad Central de Proceso 8080 no indica directamente su estado interno, transmitiendo una salida de "control de estado" durante cada uno de los mismos, sino que proporciona una salida de control directa, a utilizar por la lógica exterior.

El comienzo de cada uno de los estados, de los que hay un mínimo de tres por cada ciclo de máquina, lo determina el flanco positivo de la señal de reloj ϕ_1. Las operaciones a realizar durante cada estado tienen lugar durante la transición entre los pulsos ϕ_1 y ϕ_2.

El primer estado (T1) de cada ciclo de máquina viene definido por la señal SYNC. Esta señal se genera durante ϕ_2, con un cierto retardo (tDC) entre el flanco positivo de esta señal de reloj y el flanco positivo de SYNC, existiendo, asimismo, este retardo entre el siguiente pulso ϕ_2 y el flanco negativo de SYNC. La información correspondiente al estado en curso se transfiere a las líneas D0-D7 en el intervalo entre las señales ϕ_2.

El flanco positivo de ϕ_2 durante el estado T1 carga, asimismo, las líneas de dirección del procesador (A0-A15). Esta carga de la información se realiza con un cierto retardo (tDA) respecto a ϕ_2, y permanece en las líneas hasta que llega el primer pulso ϕ_2 en el estado T3, tiempo suficiente para tomar los datos leídos desde la memoria.

Una vez que el procesador ha enviado una dirección de memoria, ésta, y en función del tiempo de acceso a la misma, tiene la posibilidad de efectuar una demanda de espera (WAIT). Cuando ésta se produce, el procesador pone a cero la línea READY, antes de que aparezca el pulso ϕ_2 en el estado T2 o TW. Mientras esta línea permanece a cero, el procesador cesa en sus operaciones, dando tiempo a que la memoria responda enviando los datos colocados en la dirección solicitada. El cese de las operaciones del procesador corresponde a su entrada en el estado adicional TW, que siempre tendrá lugar, caso de producirse, después de T2. Al propio tiempo, el procesador indica este estado enviando una señal WAIT. El paso de cero a uno de esta línea está sincronizado con el flanco positivo de ϕ_1, y tiene un cierto retardo (tDC) respecto a la entrada del procesador en el estado TW. Figura 5-26.

Un tiempo de espera puede tener una duración indefinida, puesto que el procesador permanece en esta condición hasta que la línea READY se pone de nuevo a uno. Este paso siempre tendrá lugar antes del flanco negativo de ϕ_2 a fin de tener una garantía de que el paso hacia el estado siguiente se realiza adecuadamente. El ciclo de máquina proseguirá con el flanco positivo de ϕ_1. Por todo ello, la duración del período de espera será siempre un múltiplo entero de estados TW, y por tanto, de períodos de reloj.

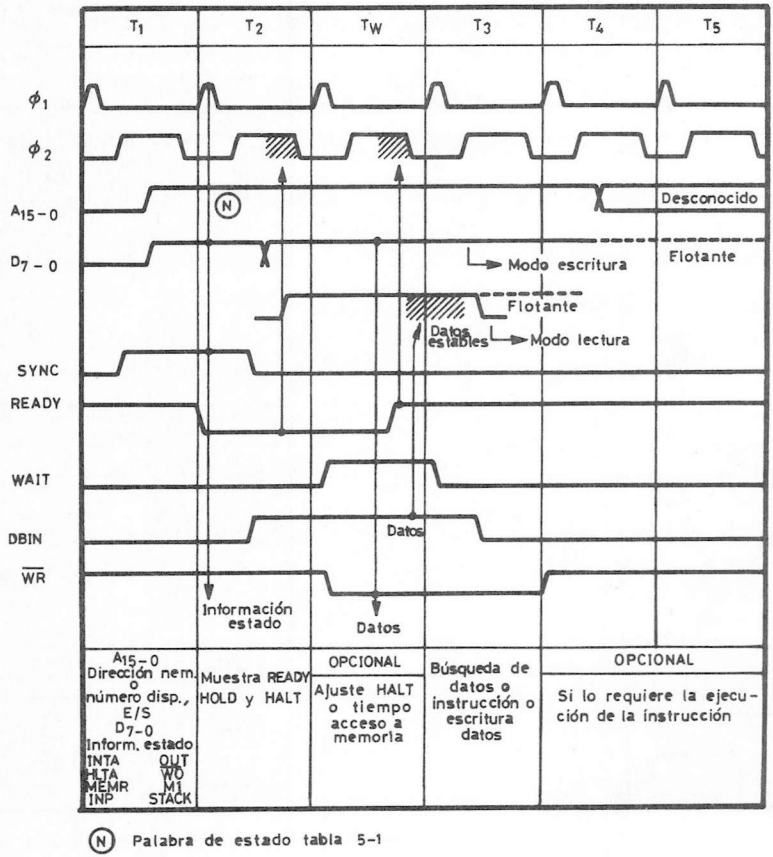


Fig. 5-26.– Ciclo de Instrucción Básico del 8080.

El estado siguiente es T3, cuyas operaciones dependen del tipo de ciclo de máquina que se está ejecutando. Si este ciclo es de búsqueda (FETCH), el procesador interpreta los datos del bus como una instrucción. Si el ciclo es de LECTURA DE MEMORIA o LECTURA DE STACK, los datos del bus se toman como una palabra de datos. Si lo que se está ejecutando es un ciclo de ESCRITURA EN MEMORIA, el procesador coloca sus datos en este bus, y por último, si el ciclo es de ENTRADA/SALIDA, el procesador puede recibir o transmitir datos desde/o a un dispositivo periférico, según se trate de una instrucción de entrada o una de salida.

A continuación vamos a determinar el proceso que sigue el procesador en una operación de entrada de datos, cuyo diagrama en el tiempo puede observarse en la figura 5-27. El flanco positivo de ϕ_2 durante el estado T2 borra el contenido de las líneas de datos del procesador, preparándolas para recibir la nueva información. Esta debe permanecer en las mismas desde unos instantes (tDS1) antes del flanco negativo de ϕ_2 en el estado T3, así como durante el tiempo de "data hold" que sucede al flanco positivo de ϕ_2. Esta información, procedente de la memoria o cualquier otro dispositivo exterior, se transferirá al procesador en el estado T3.

Durante la operación de entrada de datos al procesador, y para evitar que tenga lugar más de una transferencia desde cualquier dispositivo externo, el 8080 genera la señal DBIN, inhibiendo la transferencia desde

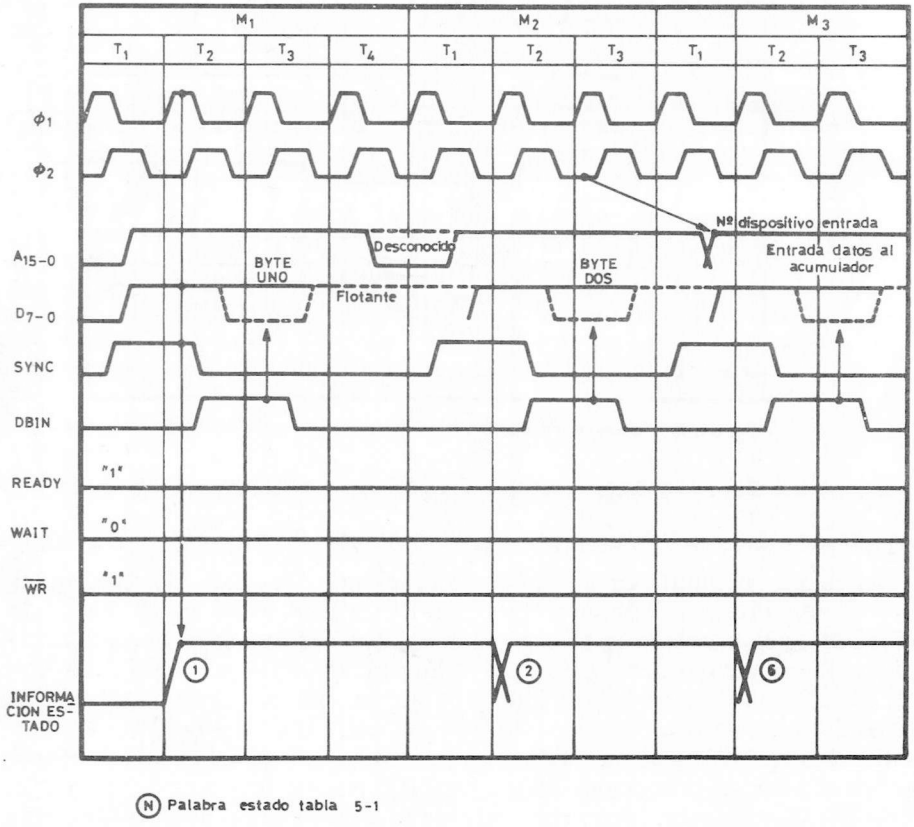


Fig. 5-27.–Ciclo de Instrucción de Entrada.

cualquier otro elemento. Esta línea puede activarse en los ciclos de máquina de FETCH, LECTURA DE MEMORIA, LECTURA DE STACK o INTERRUPCION.

Se pone a uno con el flanco positivo de ϕ_2 durante el estado T2 y cambia de polaridad con el mismo flanco del estado T3. En el caso de que existiera un estado TW entre T2 y T3 (tiempo de espera de acceso), la señal DBIN se mantendrá, por tanto, durante uno o más períodos de reloj.

Otra operación típica es la de salida de datos del procesador en un ciclo de máquina. Estos datos deben enviarse desde la CPU hacia la memoria o dispositivos periféricos. El flanco positivo de ϕ_2 en el estado T2 borra el contenido de las líneas de datos de la CPU y carga en las mismas los datos, que deben enviarse al exterior. Estos datos permanecen en las líneas el resto del ciclo de máquina hasta que son reemplazados por los de información de estado en el estado T1 del siguiente ciclo.

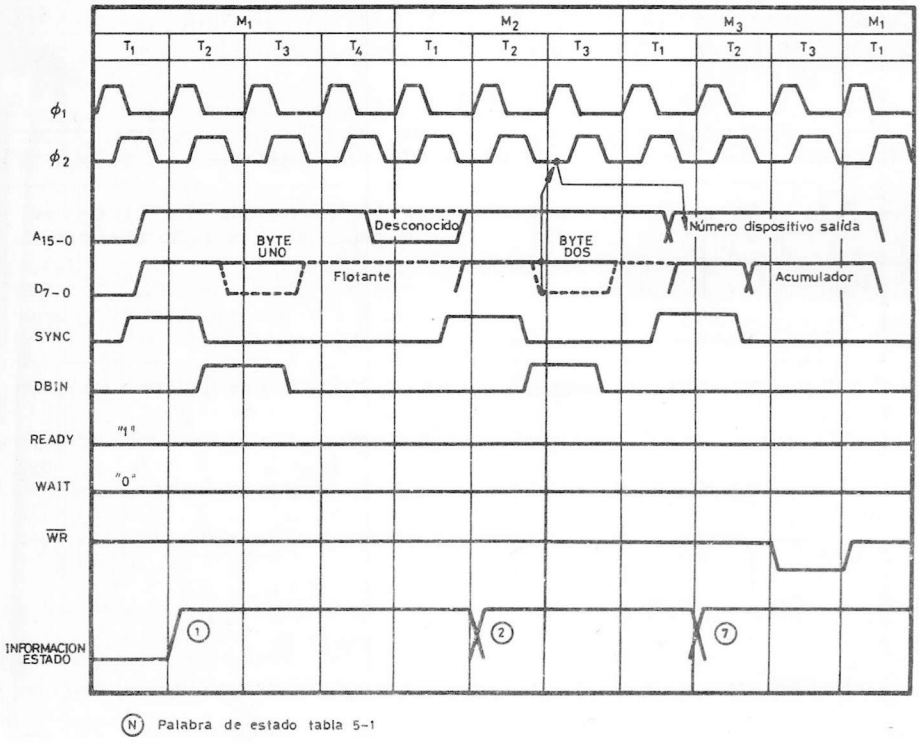


Fig. 5-28.—Ciclo de Instrucción de Salida.

Durante un ciclo de salida, la señal de READY debe estar a uno. Si esto no ocurre, el procesador entra en un estado TW después del estado T2, manteniéndose los datos en las líneas, y no continuándose el proceso hasta que la línea READY pasa de nuevo a uno.

Para sincronizar la transmisión de datos, el 8080 genera la señal WR, mientras tiene lugar la salida de los mismos del procesador. Esta señal puede activarse durante los ciclos de ESCRITURA EN MEMORIA y SALIDA, yendo sincronizado el flanco negativo de la misma con el primer flanco positivo de ϕ_1 después del estado T2, con un cierto retardo (tDC). Esta señal permanece baja hasta el primer flanco positivo de ϕ_1, después del estado T3. Como ocurría en el caso anterior con la señal DBIN, WR permanecerá invariable durante el posible estado TW entre T2 y T3. (Ver la figura 5-28.)

Todos los ciclos de máquina constan, como mínimo, de los estados T1, T2 y T3 descritos. Si el procesador debe realizar una espera para obtener una respuesta desde la memoria o desde el exterior, el ciclo de máquina constará además de uno o más estados TW. Durante los tres estados básicos, los datos se transfieren de/o hacia el procesador.

ESTADO	ACTIVIDADES ASOCIADAS
T1	Se coloca en el bus de dirección una dirección de memoria o el número de un dispositivo de entrada/salida; la información de estado se coloca en el bus de datos.
T2	La CPU verifica las entradas READY y HOLD para una instrucción de Alto.
TW (opcional)	El procesador entra en un estado de espera (wait), si la línea READY está a cero o si se ha ejecutado una instrucción de ALTO.
T3	Se introduce en la CPU, procedente del bus de datos, un byte de instrucción (ciclo de máquina de FETCH), un byte de datos (LECTURA DE MEMORIA, LECTURA DE STACK) o una instrucción de interrupción (ciclo INTERRUPCION), o bien, se saca al bus de datos, procedente de la CPU, un byte de datos (ESCRITURA DE MEMORIA, ESCRITURA DE STACK o SALIDA).
T4 T5 (opcional)	Los estados T4 y T5 únicamente tienen lugar, si una instrucción los requiere para su ejecución. Si no es así, la CPU puede saltarse uno o ambos. Unicamente se utilizan para operaciones internas del procesador.

Tabla 5.2.–Definiciones de Estado.

Transcurrido el estado T3, sin embargo, resulta más difícil generalizar, puesto que los siguientes estados, que pueden producirse, T4 y T5, dependen de que los requiera una instrucción determinada, y un ciclo de máquina en la ejecución de esta instrucción. Una vez que el procesador ha ejecutado los estados T4 y T5, si los hubiera, después de los estados T1, T2, T3, se ha completado el ciclo de máquina en curso, y se procede a ejecutar el estado T1 del siguiente ciclo.

TEORIA DE OPERACIONES DE INTERRUPCION

El procesador 8080 tiene la capacidad de responder a demandas externas de interrupciones, indicadas por un dispositivo periférico, poniendo a uno la línea de interrupción del procesador (INT).

La demanda y consiguiente entrada de interrupción es asíncrona, de tal forma, que puede producirse en cualquier momento de un ciclo de instrucción.

La lógica interna del procesador reconoce esta demanda y la ejecuta en el momento adecuado, sincronizada con las señales de reloj. Tal como se muestra en la figura 5-29, una demanda de interrupción llega en el momento en que la línea de actuación de interrupciones (INTE) está a uno, punto que coincide, asimismo, con la parte positiva de la señal de reloj ϕ_2, lo cual hace que se active la báscula interna de interrupciones, operación que tiene lugar durante el último estado del ciclo de instrucción, que se está ejecutando, cuando tiene lugar la demanda. De esta forma, se tiene la seguridad de que la instrucción, en curso de ejecución, se completa antes de procesarse la interrupción.

El ciclo de interrupción, que tiene lugar una vez que ha sido aceptada la demanda de la misma, es muy similar al ya comentado ciclo de búsqueda, puesto que se transmite el bit de estado M1 durante el tiempo de actuación de SYNC. No obstante, en este caso, va acompañado del bit de estado INTA, que determina la existencia de la demanda. El contenido del contador de programa se coloca en las líneas de dirección de la CPU durante T1, pero no se incrementa durante el ciclo de interrupción, consiguiendo que se conserven los estados previos a la interrupción, para restaurarlos, una vez que continúa la ejecución del programa en curso, cuando aquélla ha sido procesada.

El ciclo de máquina de búsqueda (fetch) no puede distinguir si la operación, que se está realizando, es una interrupción o un ciclo ordinario, y por tanto, no realiza ninguna operación específica, pues atañe a la lógica periférica el saber que, durante el estado T3, en el bus de datos de la CPU se colocan los ocho bits de una instrucción de interrupción.

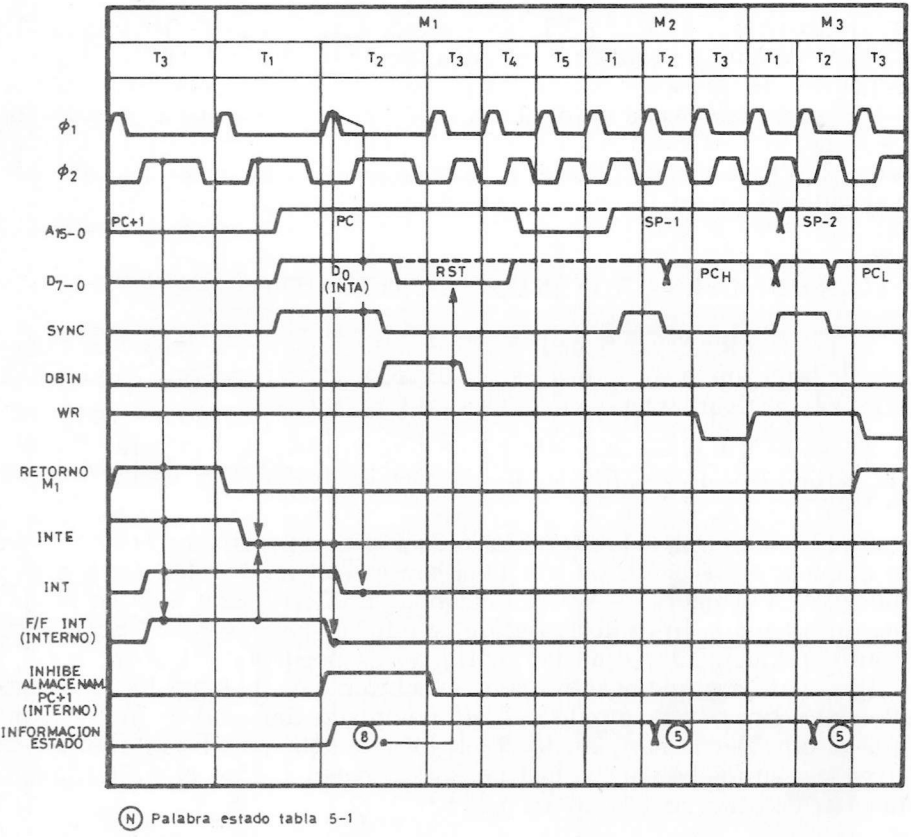


Fig. 5-29.–Ciclo de interrupción.

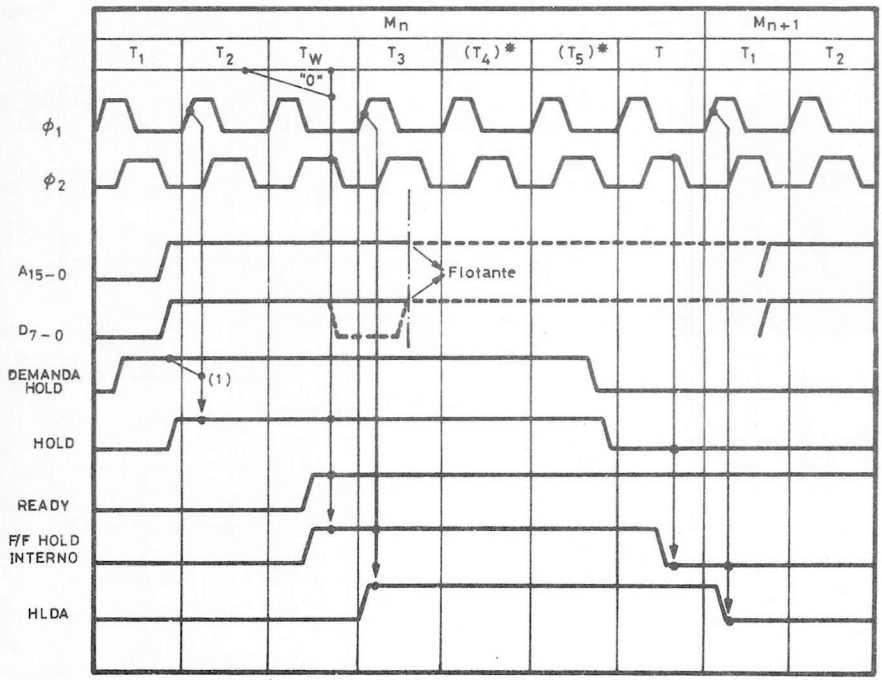
Normalmente, el bus de datos de la memoria al procesador, se inhibe a fin de que el dispositivo, que provoca la interrupción, tenga a su disposición el bus sin interferencias.

Dentro del conjunto de instrucciones del 8080 existe una de un byte, que facilita en gran manera el procesamiento de las interrupciones, pues aunque, en principio, la instrucción de "llamada" parece la idónea para este cometido, ocupa tres bytes de memoria. La instrucción, a que hacemos referencia, es la de "reinicio" (RST), en cuyo código hay un campo de tres bits, que permite al dispositivo de interrupción realizar directamente una operación de llamada a una entre ocho posiciones de memoria determinadas. La dirección decimal de estas posiciones es: ϕ, 8, 16, 24,

32, 40, 48 y 56. Cualquiera de ellas puede utilizarse para almacenar las ocho primeras instrucciones de una rutina de servicio del dispositivo de interrupción. Al igual que en las instrucciones de llamada, cada vez que se realiza una operación de RST, se guarda en el stack el contenido del contador de programa.

TEORIA DE OPERACIONES DE HOLD

La Unidad Central de Proceso 8080 está preparada para admitir operaciones de acceso directo a memoria, puesto que, aplicando una señal de HOLD en la entrada correspondiente del procesador, un dispositivo exterior puede hacer que aquél suspenda sus operaciones y le ceda el control de los buses de dirección y de datos. Cuando el procesador acepta la de-



(1) Ver características eléctricas

* Las operaciones de T_4 y T_5 pueden realizarse internamente

Fig. 5-30.–Operaciones de Hold (Modo Lectura).

manda de Hold, pone a uno su línea de salida HLDA, permitiendo operaciones de transferencia de datos a la memoria sin su intervención.

Al igual que ocurría con la interrupción, la entrada de HOLD va sincronizada internamente, aceptándose la demanda antes del flanco positivo de ϕ_2.

En las figuras 5-30 y 5-31 pueden verse en el tiempo la secuencia de operaciones de HOLD, así como el retardo existente entre la demanda de la misma, asíncrona, y la sincronización de la misma. Cuando ésta tiene lugar, coincidente ademas con la señal READY, se dispara la salida HLDA, al llegar al siguiente flanco positivo de la señal de reloj ϕ_1.

Como se ha dicho, al ser aceptada por el procesador una demanda de HOLD, los buses de dirección y datos quedan desconectados del mismo, haciéndose cargo de ellos el dispositivo externo, que produce la demanda. Esta circunstancia se produce ligeramente después de que ha sido re-

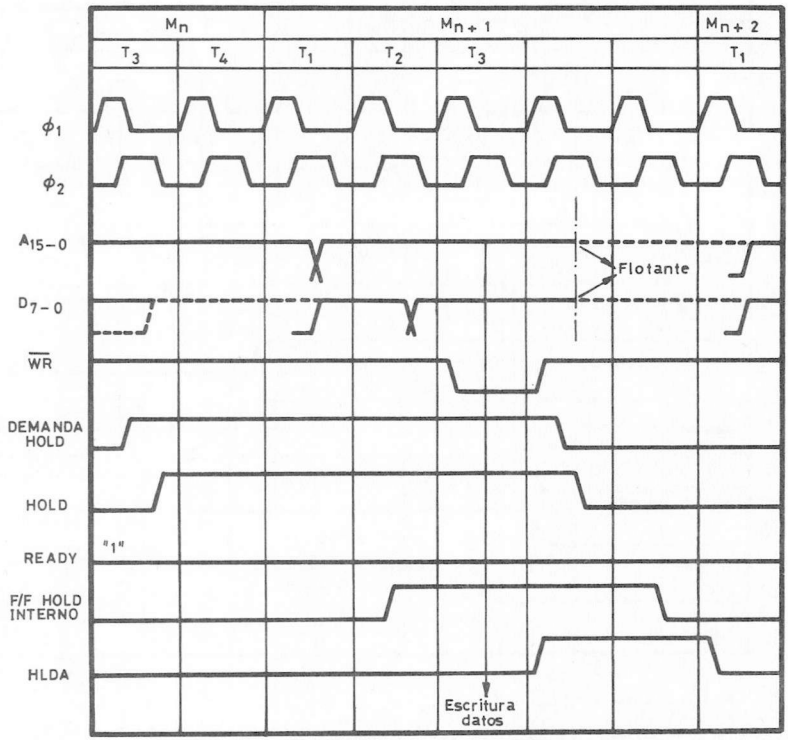


Fig. 5-31.–Operaciones de Hold (Modo Escritura).

conocida y aceptada la demanda, hecho que tiene lugar al principio del estado T3, en el caso de que esté en curso un ciclo de lectura o de entrada, o al principio del siguiente estado a T3, si esto no ocurre. En ambos casos, sin embargo, la señal HLDA se pone a uno con un retardo (tDC) respecto al flanco positivo de la señal ϕ_1 correspondiente, y los buses quedan desconectados un instante después del flanco positivo de ϕ_2, tal como puede observarse en las citadas figuras.

Una vez que los buses quedan desligados del procesador, externamente éste parece haber suspendido todas sus operaciones. Esto, no obstante, puede no ser cierto, puesto que, en el caso de que la aceptación de la demanda se produzca en el estado T3, y el procesador esté ejecutando un ciclo de máquina, que requiera para su finalización cuatro o más estados, éstos dos estados adicionales pueden realizarse internamente, de forma simultánea a la transferencia exterior de datos, dando al sistema una mayor velocidad y eficiencia.

La salida del procesador del estado de HOLD se realiza en forma muy similar a como se entra en él. La demanda finaliza, asimismo, de una forma asíncrona, una vez que el dispositivo exterior da por terminada la transferencia.

La señal de salida HLDA vuelve a su primitivo estado de cero con el flanco negativo del siguiente pulso ϕ_1, produciéndose, a continuación, la ejecución normal de las operaciones, correspondientes al siguiente ciclo de máquina.

TEORIA DE OPERACIONES DE HALT

Cuando se ejecuta una instrucción HLT, la CPU entra en un estado de "alto" en sus operaciones, tras el estado T2 del siguiente ciclo de máquina, tal como puede observarse en la figura 5-32. Para salir de este estado, el 8080 dispone de tres procedimientos.

- Mediante una operación de puesta a cero (RESET), consistente en enviar un nivel bajo por la línea RESET, lo que hace que el 8080 vaya al estado T1, borrándose, asimismo, el contenido del contador de programa. (Figura 5-34.)
- A través de una operación de HOLD, descrita anteriormente, poniendo la línea HOLD a uno. Una vez finalizada la operación, no obstante, y al quedar de nuevo a cero la línea HOLD, el 8080 vuelve a entrar en su anterior estado de alto al llegar el primer pulso ϕ_1.
- Con una interrupción, consistente en poner a uno la señal de entrada al procesador INT, siempre y cuando esté actuada la línea INTE,

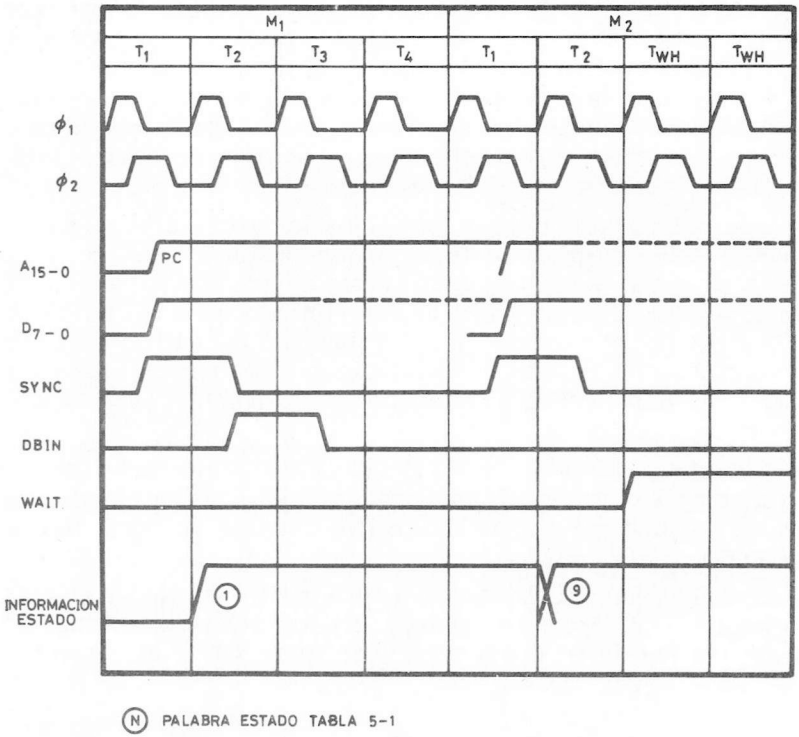


Fig. 5-32.–Operaciones de Alto.

que dará lugar a que el procesador salga del estado de alto, al aparecer el flanco positivo del siguiente pulso ϕ_1. (Figura 5-35.)

El diagrama de flujos de la figura 5-35 muestra todos estos procesos.

PUESTA EN MARCHA DE LA CPU 8080

Cuando se aplica potencia inicialmente a la CPU 8080, ésta comienza a operar inmediatamente. No obstante, como quiera que el contenido de su contador de programa, puntero de stack y otros registros toman un valor aleatorio indeterminado, es necesario siempre realizar, a continuación de la aplicación de potencia, una operación de puesta a cero (RESET).

Al realizar esta operación, de duración, como mínimo de tres períodos de reloj, se pone a cero el contenido del contador de programa del proce-

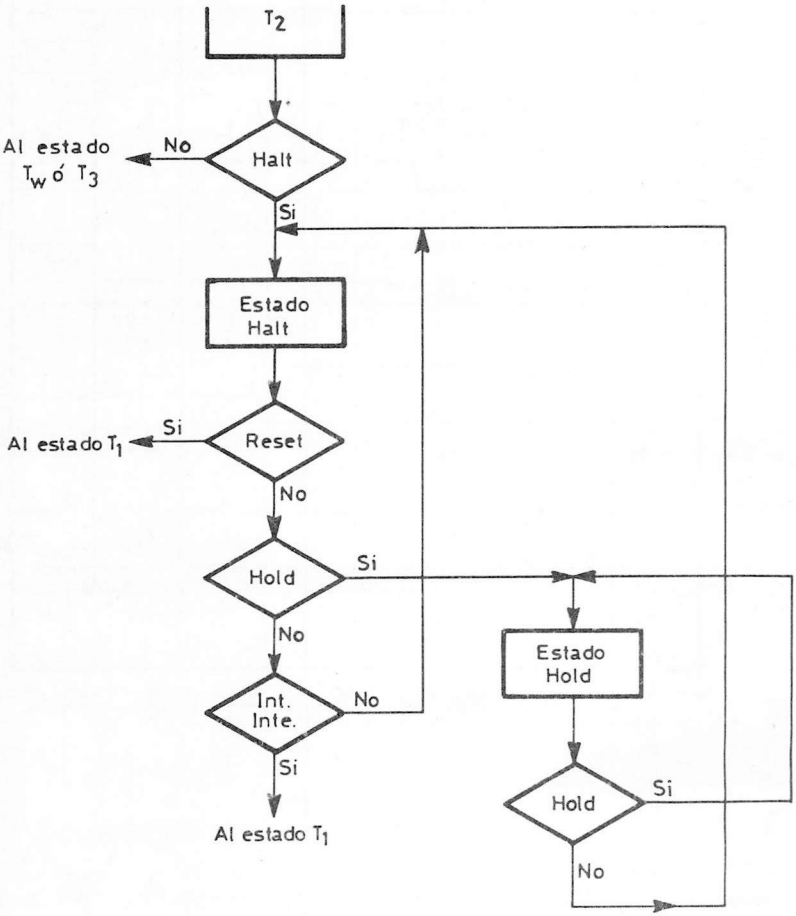
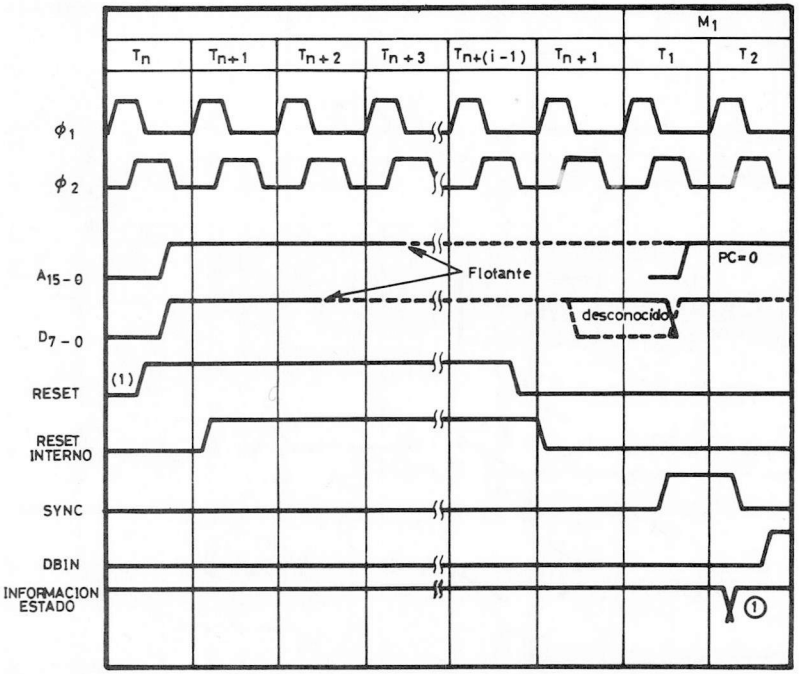


Fig. 5-33.–Diagrama de Flujo de la Secuencia Halt.

sador, por lo que la ejecución del programa comenzará en la posición de memoria cero.

Si el sistema requiere una espera del procesador hasta un momento de inicio de las operaciones determinado, se colocará una instrucción de alto (EI, HLT) en las dos primeras posiciones de memoria, de tal manera que el procesador quedará en este estado, una vez realizada la operación de puesta a cero. Deberá entonces utilizarse una interrupción manual o automática para su puesta en marcha.



(1) Cuando se activa la señal RESET, todas las señales de control de salida deben ponerse a cero inmediatamente o algunos períodos mas tarde. Esta señal, debe estar activa un mínimo de tres períodos de reloj.

(N) PALABRA DE DATOS TABLA 5-1

Fig. 5-34.–Puesta a Cero.

En otros sistemas, sin embargo, el procesador puede comenzar a ejecutar el programa contenido a partir de la posición cero de memoria, inmediatamente después de la operación de puesta a cero (RESET), teniendo en cuenta que éste no afecta ni a los bits de condición ni a los registros de trabajo (acumulador, registros, o puntero de stack), por lo que el contenido de los mismos será indefinido, hasta que el programa especifique los datos que deben contener.

NOTAS

(1) El primer ciclo de memoria (M1) es siempre una operación de búsqueda de instrucción (FETCH), identificándose durante el mismo el primer (o único) byte, que contiene el código de la operación.

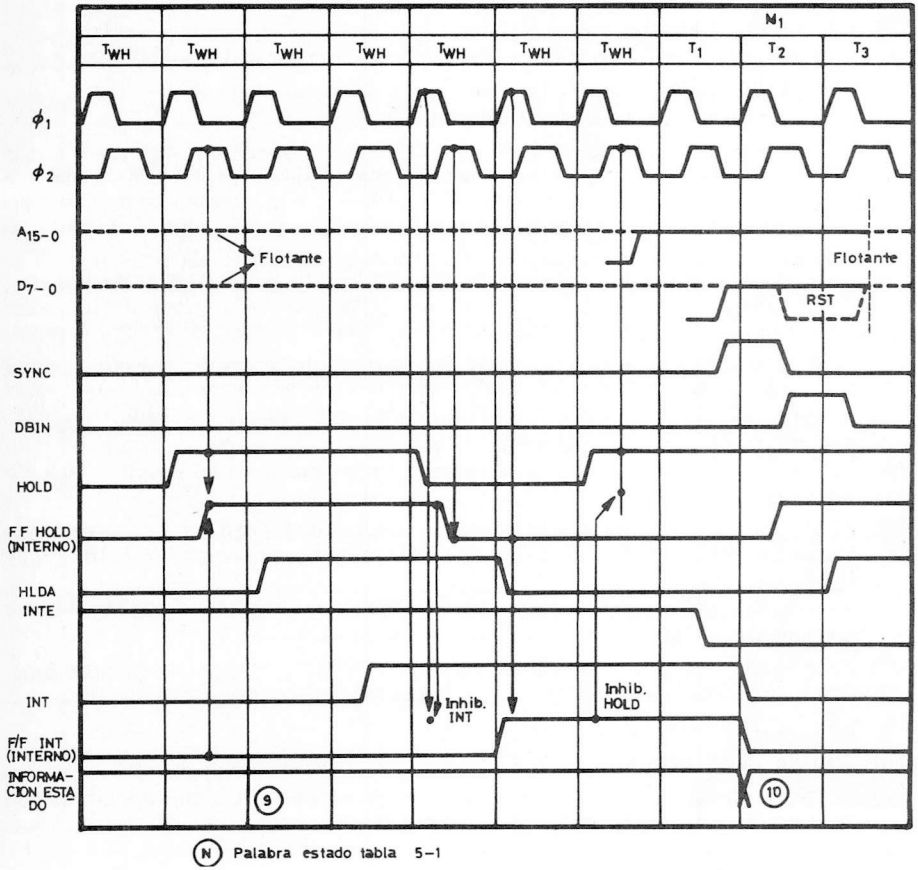


Fig. 5-35.–Relación entre Hold e Int en el Estado Halt.

(2) Si la señal de entrada READY, procedente de la memoria, no está a uno durante el estado T2 de cada ciclo de memoria, el procesador entrará en un estado de espera (WAIT), hasta que la línea READY alcance dicha polaridad.

(3) Cuando aparecen los estados T4 y T5, se refieren únicamente a operaciones totalmente internas a la CPU. Durante los mismos, el bus interno puede tomarse como bus de datos. Una "X" significa que el estado está presente, pero que únicamente se utiliza para operaciones internas, como puede ser la decodificación de la instrucción.

(4) Unicamente pueden hacer referencias a los pares de registros B (registros B y C) y D (registros D y E).

(5) Este estado se salta.

(6) Son sub-ciclos de lectura de memoria, en los cuales se lee una instrucción o una palabra de datos.

(7) Es un sub-ciclo de escritura en memoria.

(8) En esta instrucción no es necesaria la señal READY durante el segundo y tercer sub-ciclos (M2 y M3), aceptándose en éstos la señal HOLD. Asimismo, durante los mismos no se genera la señal SYNC, siendo M2 y M3 necesarios para una suma interna de un par de registros, no pudiéndose hacer referencia a la memoria.

(9) Los resultados de estas operaciones aritméticas, lógicas o de rotación, no se guardan en el acumulador hasta el estado T2 del siguiente ciclo de instrucción, o sea, hasta que la siguiente instrucción ha sido buscada e identificada. Este proceso se realiza para dar una mayor velocidad al proceso de las operaciones.

(10) Si el valor de los cuatro bits menos significativos del acumulador es mayor que 9, ó si el bit de arrastre auxiliar está a uno, se suma 6 al acumulador. Si el valor de los cuatro bits más significativos es ahora mayor que 9, ó si el bit de arrastre está a uno, se suma 6ϕH al acumulador.

(11) Representa el primer sub-ciclo (búsqueda de instrucción) del próximo ciclo de instrucción.

(12) Si la condición es verdadera, el contenido del par de registros WZ aparece en las líneas de dirección (D0-D15) en lugar del contenido del contador de programa (PC).

(13) Si la condición es falsa, se saltan los sub-ciclos M4 y M5, procediéndose inmediatamente a la búsqueda (M1) de la siguiente instrucción.

(14) Si la condición es falsa, se saltan los sub-ciclos M2 y M3, procediéndose inmediatamente a la búsqueda (M1) de la siguiente instrucción.

(15) Sub-ciclo de lectura de stack.

(16) Sub-ciclo de escritura de stack.

(17) A continuación se detalla el contenido de los tres bits CCC, en función de que se cumplan o no las condiciones de los bits de condición:

CONDICION	CCC
NZ – no cero (Z = 0)	000
Z – cero (Z = 1)	001
NC – no arrastre (CY = 0)	010
C – arrastre (CY = 1)	011
PO – paridad impar (P = 0)	100
PE – paridad par (P = 1)	101
P – signo positivo (S = 0)	110
M – signo negativo (S = 1)	111

(18) Sub-ciclo de entrada/salida. Los ocho bits de selección de puerta aparecen duplicados por las líneas de dirección: 0-7 (A0-A7) y 8-15 (A8-A15).

(19) Sub-ciclo de salida.

(20) El procesador permanecerá inhibido hasta que acepte una interrupción, una puesta a cero (RESET) a un HOLD. Cuando se acepta una demanda de HOLD, el procesador entra en este modo, volviendo al estado de alto cuando aquel finaliza. Después de aceptar una puesta a cero, el procesador pasa a ejecutar el programa colocado, a partir de la posición de memoria cero. Cuando se acepta una interrupción, el procesador ejecuta la instrucción colocada en el bus de datos (generalmente una instrucción de reinicio RST).

MICROPROCESADOR INTEL 8085

Intel produjo el primer microprocesador de 8 bits, el 8008, implementado con tecnología MOS de canal P y encapsulado en un chip dual in line (DIP). El ciclo de instrucción de este microprocesador tenía una duración de 20 microsegundos y precisaba de unos 60 componentes para formar un sistema.

A finales del año 1973, Intel ofreció el nuevo microprocesador 8080, fabricado con tecnología MOS de canal N, en un chip de 40 pines en cápsula tipo DIP. En este microprocesador la duración de un ciclo era de 2 microsegundos y el número de componentes, que se precisaba para estructurar un sistema, era de aproximadamente 30.

La versión 8080A aparecida en 1975 reduce el ciclo de instrucción a 2,3 microsegundos, y el número de componentes para completar un sistema a unos 12.

Finalmente, en 1977, el microprocesador 8085, de la misma casa, proporciona, con el mismo ciclo de instrucción de 8080A, un sistema mínimo a base solamente de 3 componentes, requiriendo una tensión única de alimentación de +5 V y estando integrado en el mismo chip, el generador de señales de reloj, el control del sistema y un conjunto muy potente de interrupciones prioritarias.

La arquitectura del 8080, descrita anteriormente, es totalmente válida y similar al del 8085, y las principales diferencias existentes entre ambos chips, pueden reducirse al analizar el diagrama de conexionado de los pines de este último, ofrecidas en la figura 5-36.

Las principales diferencias que resaltan, al comparar el diagrama de conexionado de la figura 5-36, del 8085 con el 8080, son las siguientes:

1.ª) El generador de los impulsos de reloj (clock), está integrado sobre el chip 8085, y sólo se precisa conectar un cristal o una red R-C, por sus pines 1 y 2, para estabilizar su frecuencia.

2.ª) También el microprocesador 8085 dispone, integrado en el mismo chip, el controlador de buses 8228, necesario para la adaptación del 8080.

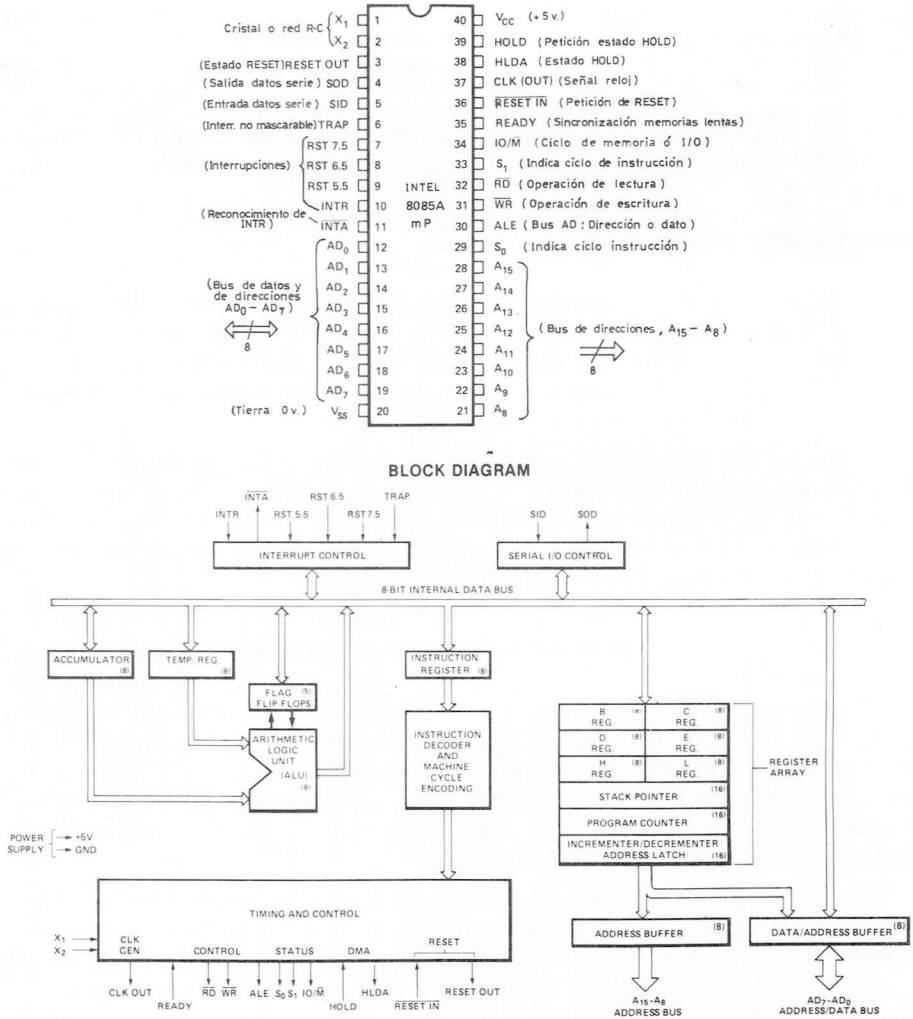


Fig. 5-36.–Diagrama de conexionado del microprocesador 8085 y arquitectura, ofrecida por INTEL.

3.ª) Precisa una alimentación única de +5 V de tensión.

Las tres características comentadas hasta el presente, quedan reflejadas esquemáticamente e la figura 5-37.

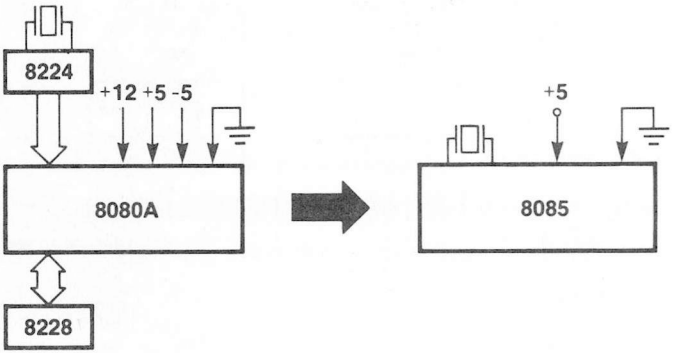


Fig. 5-37.–Similitud entre el microprocesador 8085 y el sistema preciso para realizar las mismas funciones con el 8080.

4.ª) El 8085 ofrece 4 nuevas entradas y posibilidades de interrupción, denominadas: TRAP, RST 7,5, RST 6,5 y RST 5,5.

5.ª) Dispone de líneas de entrada y salida de información, en serie (SID y SOD).

6.ª) Los 8 bits de menos peso del bus de direcciones quedan multiplexados con los 8 bits del bus de datos en los pines 12, 13, ... y 19.

7.ª) El juego de instrucciones del 8085 es similar al que posee el 8080, a excepción de 2 nuevas instrucciones específicas del 8085, que a continuación se describen:

RIM: Lee, o carga sobre el Acumulador el estado de las líneas de interrupción, sus enmascaramientos y la línea SID.

En el bit de menos peso o 1.º del Acumulador se carga el estado de la máscara para la interrupción RST 5,5. En el 2.º bit, el de la máscara de RST 6,5 y en el 3.º bit el de la RST 7,5.

En el 4.º bit del Acumulador se carga al realizar esta instrucción el flag que indica el estado del sistema de interrupciones.

En el 5.º, 6.º y 7.º bit del Acumulador la instrucción RIM carga las interrupciones pendientes si las hay de 5,5; 6,5 y 7,5 respectivamente.

Finalmente, en el 8.º bit se carga el estado de la línea SID que permite la entrada de datos en serie a la CPU 8085.

SIM: Con esta instrucción el contenido del Acumulador se utiliza para programar la máscara de interrupciones, como se ha representado gráficamente en la figura 5-39.

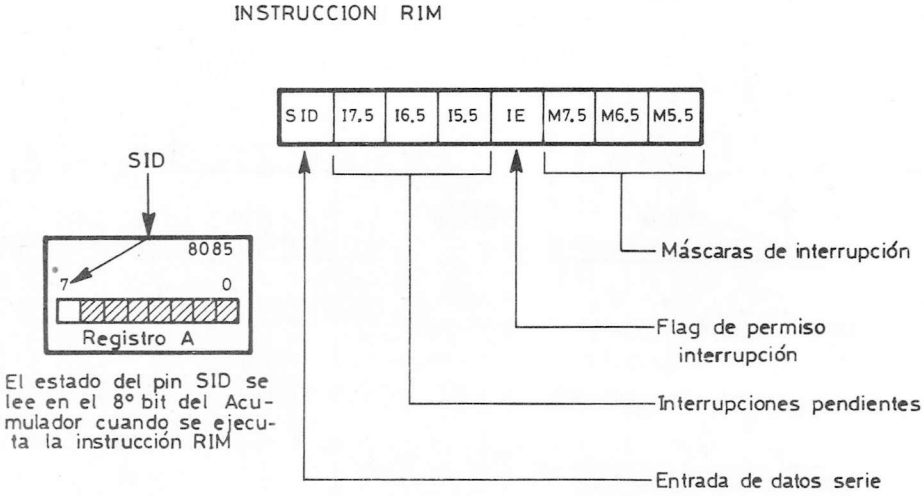


Fig. 5-38.–Contenido del Acumulador después de ejecutar la instrucción específica del 8085, RIM.

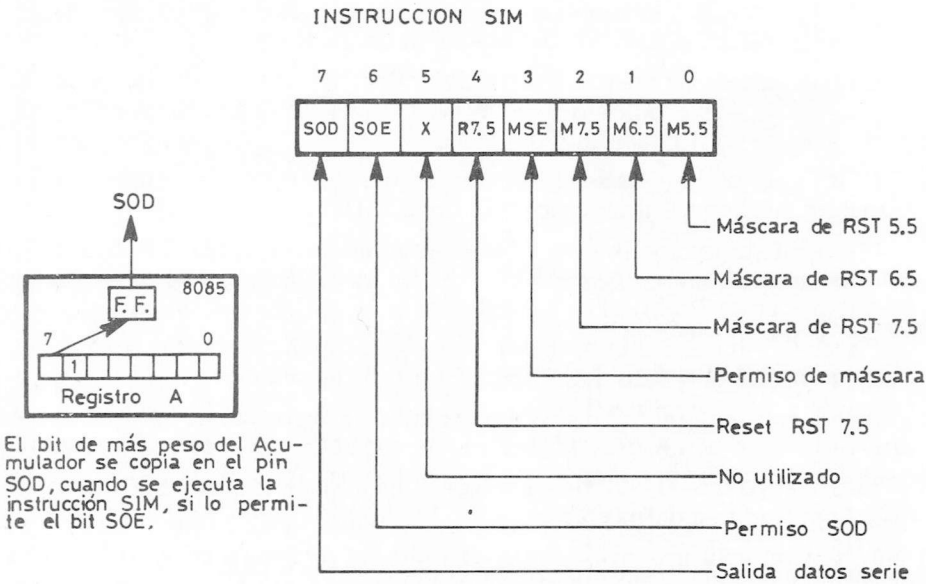


Fig. 5-39.–Contenido del Acumulador antes de ejecutarse la instrucción SIM.

Los bits 1.°, 2.° y 3.° (de menos peso) del Acumulador contienen las máscaras de las interrupciones RST 5,5; 6,5 y 7,5 respectivamente.

El bit 4.° contiene el bit para el reset de la interrupción RST 7,5. El bit 6.° no se usa. El bit 5.° contiene el estado del permiso de interrupciones. El 7.° bit permite o no la salida de datos en serie y el bit 8.° es copiado sobre la línea SO, cuando el 7.° bit está a nivel 1.

INTERRUPCIONES DEL 8085

De las características reseñadas del 8085, merece destacarse la correspondiente a las interrupciones, que potencia extraordinariamente al microprocesador.

El 8085 dispone de una línea de interrupción, INTR, similar al 8080, que permite ejecutar aquella instrucción, que se introduce a través del bus AD mediante hardware externo. Además, dispone de otras 4 líneas de interrupción, que son: TRAP, RST 5,5; RST 6,5 y RST 7,5.

La línea TRAP provoca una interrupción "no mascarable", lo que representa que siempre que se produce se ejecuta, pasando la CPU a atender una subrutina vectorizada en la posición 24_{16}. Esta línea de interrupción es activa bien por flanco o por nivel.

Las líneas RST 5,5 y RST 6,5 se activan al aplicarlas un nivel 1, que debe mantenerse, hasta que sea atendida dicha petición de interrupción. Su forma de trabajo es similar a la de la línea TRAP, pero estando vectorizada a las posiciones $2C_{16}$ y 34_{16}.

Finalmente, la línea de interrupción RST 7,5 es activa por flanco, el cual es memorizado en cuanto se produce, hasta que se proceda a la atención de dicha petición. Esta interrupción, como ya se ha indicado al explicar la instrucción SIM, específica del 8085, puede ponerse a 0 y evitar su reconocimiento.

De las 5 líneas de interrupción, disponibles en el 8085, la de mayor prioridad por ser "no mascarable" es la TRAP. A continuación, siguen en prioridad la RST 7,5, RST 6,5 y RST 5,5, por el orden citado. Por último, la línea INTR es la de menor prioridad.

En la figura 5-40 se muestra una fotografía del 8085, tomada con ayuda del microscopio.

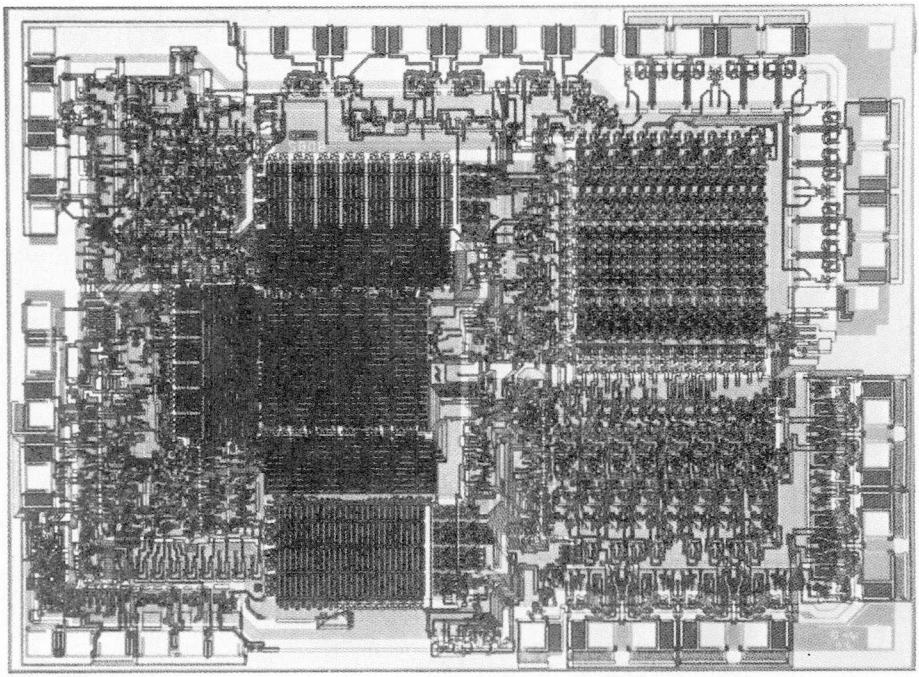


Fig. 5-40.–Fotomicrografía del microprocesador 8085. El chip es de una superficie cuadrada de 0,2 pulgadas y contiene unos 20.000 transistores.

FLUJO DE INFORMACION EN UNA INSTRUCCION

Como el 8085, por su funcionamiento y arquitectura es muy parecido al 8080, y al describir éste ya se comentaron casi todos los temas de interés, sólo se facilita a continuación, y de manera resumida, el flujo de información, que se produce en los dos tipos de microprocesadores, cuando se lleva a cabo una instrucción cualquiera.

Vamos a referirnos a la instrucción MVI OD, con la cual se pretende cargar en el Acumulador el dato, que se proporciona directamente, y que, en el ejemplo, es OD. Teniendo en cuenta que el código OP de la instrucción MVI con direccionamiento inmediato es 3E, la figura 5-41 ofrece la fase de búsqueda de esta instrucción, en la cual, la dirección de la instrucción contenida en el PC, se traslada por el bus de direcciones hasta la ROM, donde se decodifica y se obtiene el contenido de dicha posición, que supongamos es la 0200, y que tendrá almacenado el código

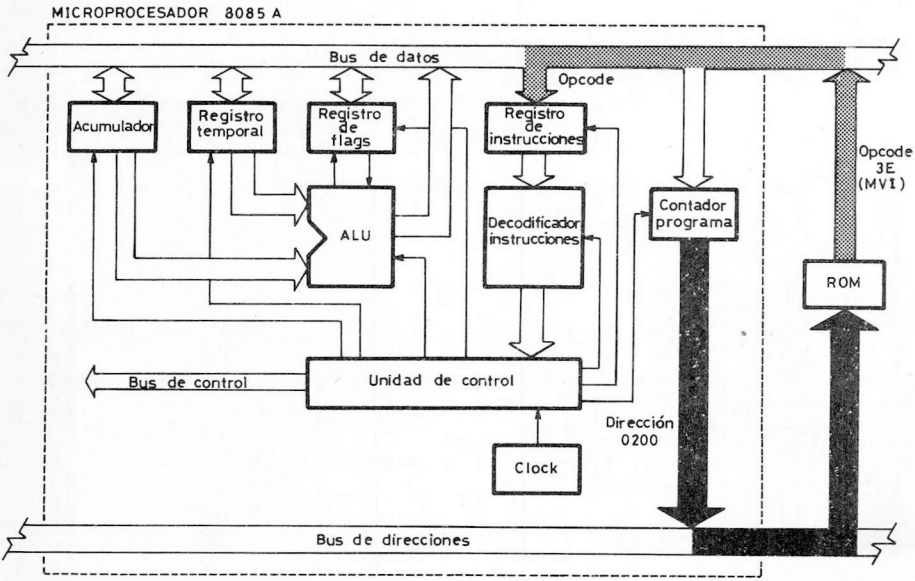


Fig. 5-41.–Fase de búsqueda de la instrucción MVI OD, contenida en las posiciones 0200 y 0201 de la ROM.

OP 3E, que por el bus de datos se lleva hasta el registro de instrucciones, donde queda almacenado.

El PC se incrementará automáticamente y quedará señalando la posición 0201, en la cual se halla el dato OD, que se desea cargar al Acumulador.

La fase de ejecución comienza decodificando el código OP 3E por el Decodificador de Instrucciones. La decodificación se realiza por comparación en una ROM de la CPU, que contiene el microcódigo o microprograma para cada instrucción, y que indica a la CPU, por medio de las señales de control, lo que debe realizar para ejecutar la instrucción decodificada. En nuestro ejemplo, la decodificación de 3E supondrá el conocimiento de que hay que cargar al Acumulador con el byte, contenido en la siguiente dirección del programa, por lo que se enviará la señal al PC para que su contenido, el 0201, por el bus de direcciones llegue a la ROM y el contenido de dicha posición 0201, que será OD, pase por el bus de datos hasta el Acumulador, en donde se carga dicho dato. La fase de ejecución de la instrucción comentada se ofrece en la figura 5-42.

Una vez conocidos algunos microprocesadores se ofrece en la Tabla 5-2 un resumen comparativo de los más representativos.

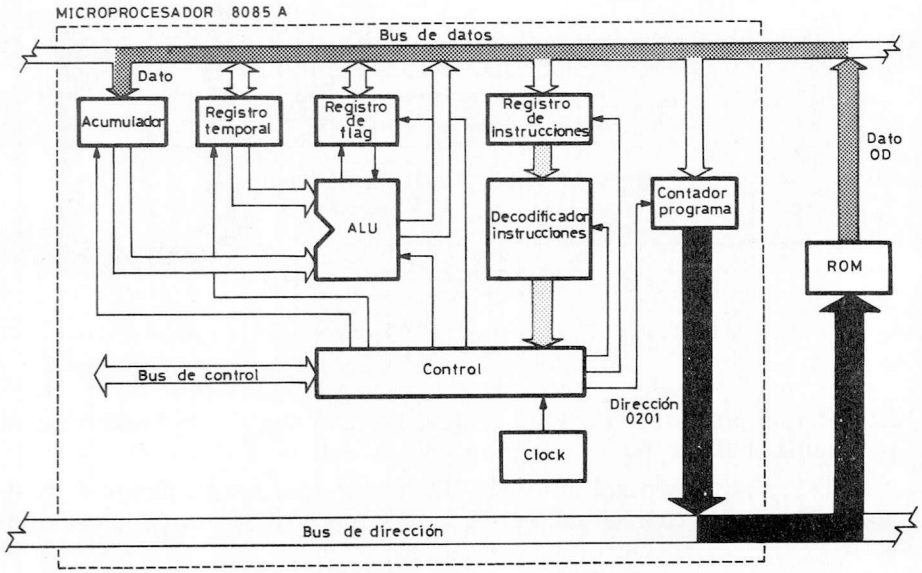


Fig. 5-42.–Fase de ejecución de la instrucción MVI OD.

CARACTERISTICAS DE MICROPROCESADORES

Propiedades / Fabricante	Modelo	Tecnología	Capacidad de memoria	Tamaño de la palabra (bits)	Número de Instrucciones	Ciclo de la Instrucción	Registros Internos	Flags de Estado	Alimentación	Interrupciones	Tiempo de reacción de Inter.	Ensamblador	Otros lenguajes
Rockwell	PPS-4	PMOS	4 K	4	50	5 μs.	4		−17 V	No			
Rockwell	PPS-8	PMOS	16 K	8	≅100	4 μs.	3		−17 V	Si (3)		Si	
MOS TECH y Rockwell	R6502	NMOS	64 K	8	56	1 μs.	3	7	+5 V	Si (3)		Si	Basic
Intel	8080	NMOS	64 K	8	70	2 μs.	8		−5, +5 y 12 V	Si (1)	20 μs.	Si	PLM
Intel	8085	NMOS	64 K	8	72	1,3 μs.	8		+5 V	Si (4)	12 μs.	Si	PLM
Intel	8086	HMOS	1.024 K	16	250	0,8 μs.	16		+5 V	Si (7)	20 μs.	Si	Basic/PLM/Fort.
Intel	3001	TTL/S	512 K			0,165 μs.			+5 V	Si (2)			
Motorola	6800	NMOS	64 K	8	72	≅1 μs.	6	6	+5 V	Si (3)		Si	APL/For./Bas./Co.
Motorola	68000	NMOS	16 M	16	61	250 ns.	18		+5V	Si (2)		Si	MPL/For./Bas./Co.
Signetics	2650	NMOS	32 K	8	75	4,8 μs.	7		+5 V	Si			
National	8060	NMOS	64 K	8	46	1 μs.	3	8	+5 V	Extern. Sense A	1 inst.	Si	Basic
National	8900	NMOS	64 K	16	46	2 μs.	4	16	+5, +12 y 8 V	Si (6)	1 inst.	(Macro)	Bas./PLM/CrosE.
National	IMP-8	PMOS	64 K	8	58	4,6 μs.			+5 y −12 V	Si			
Texas	9900	NMOS	64 K	16	69	Varía	16	7	+12, +5 y −5 V	Si	12,6 μs.	Si	Bas./Pascal/For.
Texas	990	NMOS	64 K	8	58	Varía	16	7 + 16		Si	12,6 μs.	Si	Bas./Pascal/For.
Fairchild	F8	NMOS	64 K	8	67	2,3 μs.	3	5	+5 y +12 V	Si	>11 μs.	Si	
Raitheon	RP-16	Bipolar	64 K	4		1 μs.		5					

Tabla 5-2.—Tabla comparativa de características de microprocesadores.

APENDICE

MICROPROCESADOR 2650 DE SIGNETICS

ARQUITECTURA DEL MICROPROCESADOR 2650

Con la intención de dar una idea más amplia de las características esenciales, que acompañan a todos los microprocesadores, y, de esta forma, habituar al lector a manejar diferentes tipos, manteniendo las mismas directrices e idéntica filosofía, se ofrece a continuación una exposición de otro microprocesador, comercializado en España por COPRESA, en cuyas publicaciones técnicas se basa parte de esta exposición.

El microprocesador 2650 es un circuito integrado, que constituye una Unidad Central de Proceso, para tratamiento de palabras de 8 bits, fabricado con tecnología MOS de canal N, y dispuesto en una cápsula normalizada de 40 patillas.

Otras características interesantes del 2650 son:

1) Alimentación única de +5 V.
2) ALU capaz de realizar operaciones aritméticas en sistema binario y en BCD.
3) Dispone de un total de 7 registros de 8 bits, de aplicación general.
4) Puede direccionar hasta 32 K bytes, en 4 páginas de 8 K cada una.
5) Al disponer las instrucciones de diferentes longitudes, se reduce al mínimo la memoria que almacena un programa. El primer octeto de una instrucción indica el código de la operación a realizar y el modo de direccionado. Empleando la mayor parte de las instrucciones, 6 bits para los fines citados, mientras que los dos restantes determinan un registro de aplicación general.
6) La instrucción más compleja tiene una longitud de 3 octetos y una duración de ejecución de 9,6 μs, si el generador de impulsos es de la máxima frecuencia y las memorias externas tienen un tiempo de acceso menor de 1 μs. La instrucción más rápida se ejecuta en 4,8 μs.
7) Sólo precisa una simple entrada, a la que se ha de aplicar la señal de un oscilador externo.
8) Los buses o barras de direccionamiento y de datos, son de carácter triestado.

9) El sistema de interrupciones ha sido realizado con un sólo nivel y direcciones vectorizadas, es decir, el periférico que interrumpe puede indicar directamente al microprocesador cuál es la dirección de memoria, en la que se inicia la subrutina de tratamiento de la interrupción.
10) Máxima potencia de 252 mW.
11) Entradas y salidas compatibles con tecnología TTL.
12) Juego de 75 instrucciones.
13) Acoplamiento con memorias estándar.

La figura 5-43 muestra el diagrama por bloques del 2650, en donde se aprecian los elementos fundamentales y su interrelación. Como en los anteriores tipos de CPU, la figura presenta:

- Contador de programa: que envía por el bus de direcciones la dirección de la memoria, en donde se encuentra el código de la instrucción a realizar.
- Registro de dirección de operandos.
- Barras o buses de direcciones y de datos.
- Registro de instrucciones, donde se almacena la instrucción antes de enviarla al decodificador de instrucciones para su interpretación.
- Unidad de control y Secuenciador: que genera los impulsos, derivados de la interpretación de la instrucción, para su ejecución.
- ALU en conexión con diferentes registros de trabajo, entre los que se encuentra el Registro de estado del Programa.
- Memoria LIFO de direcciones de retorno de subrutina (Stack Pointer).
- Circuitos de interrupción, de I/O y otros auxiliares.

De la descripción mencionada se puede deducir, que la arquitectura de este microprocesador se asemeja mucho a la de carácter general y a la del resto de los micros estudiados.

La ejecución de una instrucción por el 2650 conlleva las siguientes fases elementales, realizadas secuencialmente:

1.ª) Enviar el contenido del Registro de Dirección de Instrucciones (Contador de Programa) al bus de direccionamiento.
2.ª) Simultáneamente, enviar las señales de control adecuadas a la memoria externa de acceso aleatorio y almacenar el contenido de la posición direccionada en el Registro de Instrucción a través del bus de datos.

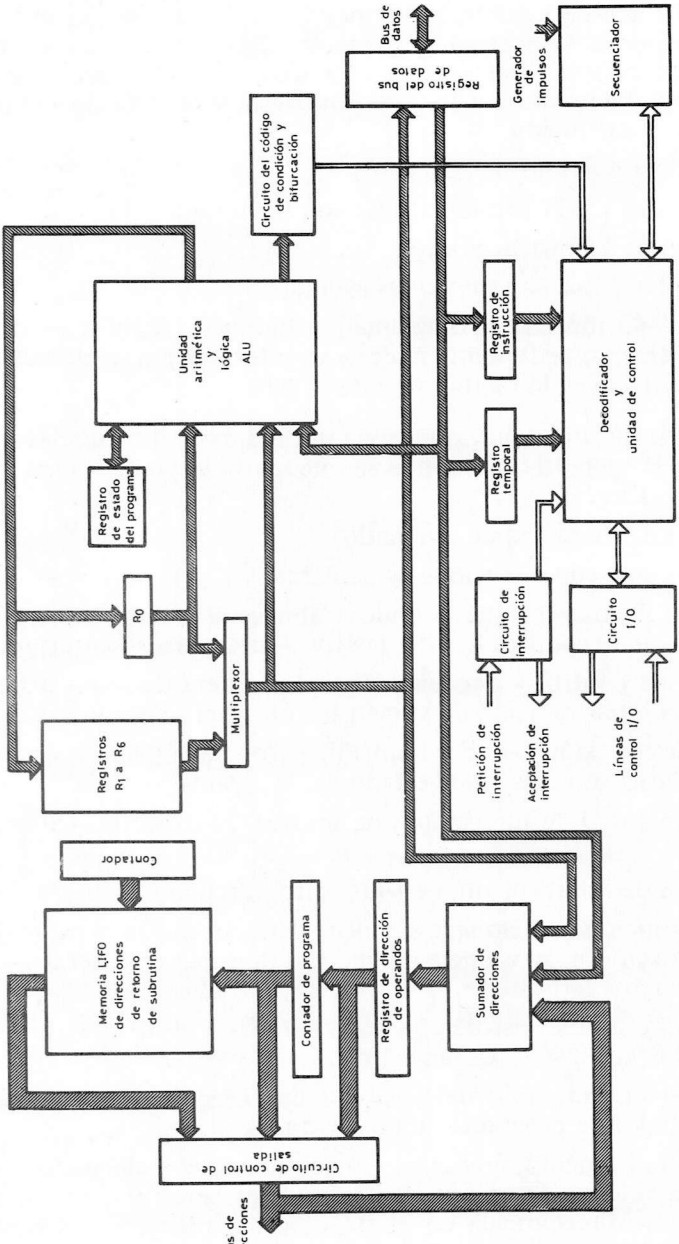


Fig. 5-43.–Diagrama por bloques del microprocesador 2650 Signetics.

3.ª) Decodificar el contenido del Registro de Instrucción para determinar el tipo de Instrucción y el tipo de direccionamiento.

4.ª) Si la instrucción necesita un operando situado en memoria, el microprocesador ha de calcular su dirección de acuerdo con las indicaciones del tipo de direccionamiento, e introducirla en el registro de dirección del operando y llevarla a las líneas del bus de dirección a través del circuito de control de salida de la dirección.

5.ª) A continuación, el microprocesador ha de dar una orden de lectura a la memoria, que hace aparecer el operando en el bus de datos y lo introduce en el Registro del bus de datos.

6.ª) Seguidamente se ejecutará la operación especificada en la instrucción y el resultado se introducirá en uno de los registros R_0 a R_6, según el estado del campo correspondiente de la instrucción.

Las señales, que entran y salen del microprocesador, se presentan agrupadas en la figura 5-44.

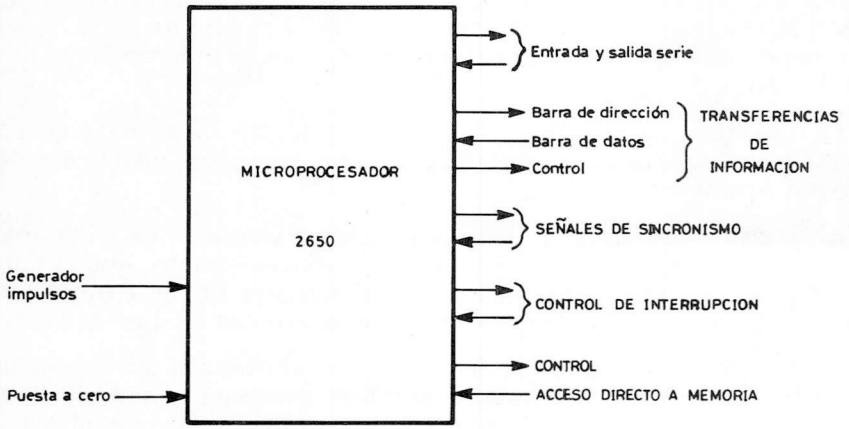


Fig. 5-44.–Agrupación de las señales del microprocesador 2650.

La función de cada bloque de la arquitectura del 2650 dentro de la operación de la CPU es la siguiente:

El Registro de Instrucción almacena temporalmente el primer octeto de cada instrucción. Su contenido se decodifica y actúa sobre la unidad

de control, que genera las señales necesarias para la ejecución de la instrucción.

El registro temporal se utiliza en algunas de las instrucciones de dos o tres octetos para almacenar parte de la dirección del operando en el direccionamiento absoluto.

La ALU realiza las operaciones lógicas, aritméticas y de transferencia con los operandos: Carga de los registros R_0 a R_7 con el contenido de una posición de memoria. Almacenamiento de los registros R_0 a R_7. Además efectúa la suma binaria, la resta binaria y lógica, operación lógica 0,0-Exclusiva, Comparación, Desplazamiento del contenido de los registros R_0 a R_7. Controla también el bit de Acarreo, el bit de Rebasamiento, el bit de acarreo entre dígitos (BCD), y los bits del Código de Condición del Registro de Estado y control de programa.

La unidad de memoria, que hay dentro del microprocesador, contiene 6 registros (R_1 a R_6), organizados en 2 grupos de 3 registros cada uno. El bit de Selección de registro (RS) del Registro de estado y control del programa permite seleccionar al grupo, con el cual se realizan las operaciones. Las transferencias de información se realizan siempre entre un registro de uno de los dos grupos y el registro R_0, que es independiente de todos los demás.

El cálculo de las direcciones en los diferentes tipos de direccionamiento se realiza mediante un Sumador de direcciones, independiente de la Unidad Aritmética y Lógica.

El Registro de Dirección de la Instrucción o Contador de Programa almacena el contenido de la dirección del próximo octeto, que el microprocesador ha de leer en la memoria y el Registro de Dirección del operando los datos y resultados parciales en el cálculo de las direcciones.

La memoria de direcciones de retorno de subrutina es del tipo secuencial LIFO de 8 niveles. En esta memoria se almacena la dirección de retorno, cuando el microprocesador realiza una instrucción de salto a subrutina. Cuando, al terminar la subrutina, se ejecuta una instrucción de retorno al programa principal, la unidad de control transfiere la información de la memoria de retorno de subrutinas al Registro de dirección de la Instrucción o Contador de Programa.

Los 8 niveles de la memoria secuencial utilizada, LIFO, permiten realizar sucesivos saltos a subrutinas en el interior de otra subrutina hasta un número máximo de 8. La memoria LIFO está asociada a un contador de 3 bits, cuyo estado indica la posición de la cual se extraerá la información, al realizar la primera instrucción de retorno al programa principal.

REGISTRO DE ESTADO Y CONTROL DEL PROGRAMA

El Registro de estado y control del programa (PSW) es un registro especial del microprocesador 2650, que contiene bits de control y de estado del programa. Su longitud es de 16 bits y está dividido en 2 octetos, que reciben los nombres de Superior e Inferior.

El Registro de estado y control del programa puede ser comprobado, cargado con un octeto, procedente de una posición de memoria, almacenado en una posición de memoria, borrado o puesto a uno los bits, seleccionados mediante instrucciones especiales. El bit de detección (Sense bit) no puede ser borrado, porque está conectado directamente al terminal 1. En la figura 5-45 se indica el significado de cada uno de los bits del registro de estado y control de programa.

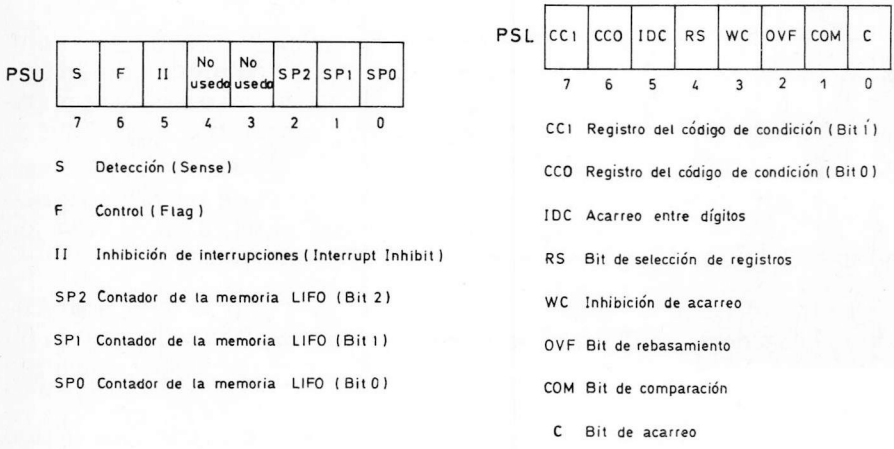


Fig. 5-45.–Bits del registro de estado y control del programa.

"Bit Sense (S)": Es un bit de detección, que memoriza el estado del terminal 1 del microprocesador.

"Bit de Control (F)": Es un biestable, cuya salida está unida al terminal 40 del microprocesador.

"Bit de inhibición de interrupción (II)": Constituido por un biestable, que, cuando se encuentra en estado uno, impide la aceptación de interrupciones por parte del microprocesador. Cuando su estado es cero, las interrupciones están desinhibidas y, al producirse éstas, automáticamente cambian a uno el estado de este biestable.

"Contador de la memoria LIFO (SP)": Tiene 3 bits (SP2, SP1 y SP0), que direccionan la memoria LIFO, que contiene las direcciones de retorno de las subrutinas. Su contenido indica la posición de la memoria LIFO, que guarda la dirección de retorno de la subrutina, que está ejecutándose.

"Registro del Código de condición (CC)": Está formado por dos biestables, cuyo estado es modificado, cuando el microprocesador realiza una instrucción de carga o cambio del contenido de uno de los registros R_0 a R_7.

Las instrucciones de comparación actúan también sobre el contenido del registro afectado.

Las instrucciones de bifurcación condicional utilizan el contenido de este registro para tomar o no la decisión de saltar.

"Acarreo entre dígitos (IDC)": En las operaciones aritméticas en código BCD natural, es, a veces, necesario conocer si se ha producido un acarreo del bit 3 al 4. El bit IDC del registro de estado y control de programa, memoriza el valor de dicho acarreo en la última suma o resta que se ha realizado.

El bit IDC memoriza también el estado del bit 5 de un registro, después de haber realizado el microprocesador una instrucción de rotación con su contenido, si el bit WC se encuentra en estado lógico uno.

"Bit de selección de registros (RS)": Existen 2 grupos de registros de aplicación general en el 2650, que contienen 3 registros cada uno. El bit RS permite seleccionar en qué grupo está el registro utilizado al realizarse una instrucción.

El registro R_0 es independiente y no pertenece a ninguno de dichos grupos.

Cuando el estado de RS es un cero lógico, se selecciona uno de los registros del grupo 0, y cuando su estado es un 1, se selecciona uno de los registros del grupo 1.

"Inhibición de acarreo (WC)": Este bit controla la ejecución de las instrucciones de suma, resta y rotación.

Cuando se realiza una instrucción de suma o resta, el resultado puede hacer cambiar el estado de los bits de acarreo (C), rebasamiento (OVF) o acarreo entre dígitos (IDC), independientemente del estado lógico del bit WC. Pero si WC es un uno lógico, el valor final del acarreo afecta al resultado de una suma o una resta, sumándose o restándose respectivamente del bit menos significativo.

Si cuando se realiza una instrucción de rotación, el bit WC se encuentra en estado lógico cero, sólo se desplazan los 8 bits del registro considerado. Por el contrario, si el bit WC se encuentra en estado lógico uno, el bit de acarreo (C) se combina con los 8 bits del registro para formar un registro de 9 bits; además, el bit de sobrepasamiento (OVF) es puesto a 1 cuando el bit 7 cambia de cero a uno o viceversa, y el bit de acarreo entre dígitos (IDC) memoriza el nuevo valor del bit 5 del registro, después de realizarse la rotación (figura 5-46).

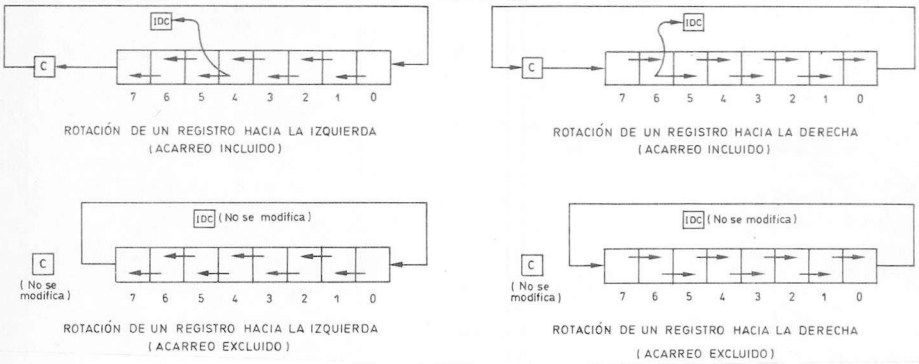


Fig. 5-46.–Operación de rotación.

"Bit de rebasamiento (OVF)": Se pone a uno al ejecutarse una instrucción de suma o resta, cuando los dos operandos tienen el mismo signo y el resultado tiene un signo diferente al de ambos. Los operandos que poseen distinto signo no pueden causar rebasamiento.

Las instrucciones de rotación del contenido de un registro también ponen en estado uno el bit de rebasamiento, cuando cambia de signo el octeto al realizar la rotación.

"Bit de control de comparación (CCM)": Determina el tipo de comparación que ejecutan las instrucciones de comparación. La comparación puede ser lógica o aritmética.

"Bit de acarreo (C)": Se pone a uno cuando se ejecuta una instrucción de resta o de suma, que produce un bit de acarreo al operar los dos bits más significativos de la ALU. El bit de acarreo es puesto a 1 por una instrucción de resta, que no produce acarreo.

El bit de acarreo puede también ser puesto a 1 o borrado por las instrucciones de rotación.

DIAGRAMA DE LOS TERMINALES DEL MICROPROCESADOR 2650

En la figura 5-47 se muestra el diagrama de conexionado del microprocesador de Signetics.

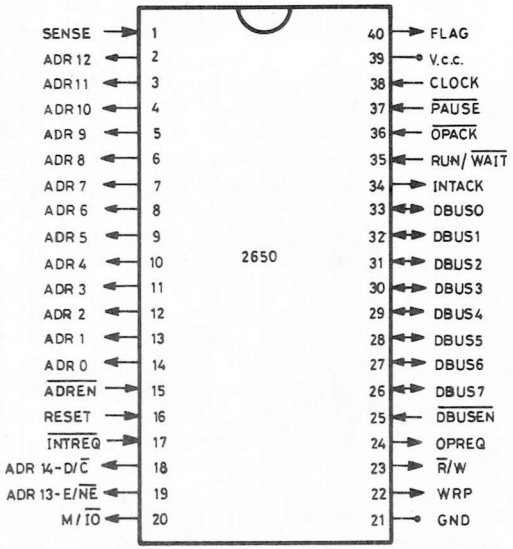


Fig. 5-47.–Diagrama del conexionado de los terminales del 2650.

A continuación, se describe la función de cada una de las patitas del circuito integrado, correspondientes a la figura 5-47.

"Terminal de puesta a cero (RESET)": Permite fijar el estado inicial interno del microprocesador. Normalmente se utiliza para fijar el estado inicial del microprocesador, al aplicarle tensión o reiniciar un programa.

Los flancos de subida y bajada de la señal, aplicada al terminal RESET, pueden ser asíncronos con respecto a los impulsos del generador.

"Terminal del generador de impulsos (CLOCK)": A esta patilla se ha de aplicar la señal de un generador de impulsos, que sincronizará la ejecución en secuencia de las instrucciones. Un ciclo del microprocesador necesita 3 períodos del generador de impulsos.

"Terminal de Paro (PAUSE)": Permite temporalmente inhibir la ejecución de un programa. Cuando la señal aplicada a este terminal pasa del estado uno al cero, el microprocesador termina la instrucción que está

realizando y se detiene. Cuando dicha señal vuelve al estado lógico uno, continúa la ejecución del programa a partir de la instrucción en la que se detuvo.

Si se produce una Petición de Interrupción mientras el terminal de Paro se encuentra en estado cero, la interrupción es aceptada y atendida después de realizada la primera instrucción, al pasar al estado uno la señal de Paro.

"Terminal de Petición de Interrupción (INTREQ)": Permite a los dispositivos externos al microprocesador, o periféricos, interrumpir el programa, que se está realizando en un determinado instante, y lograr que el microprocesador les preste atención. Cuando la señal INTERQ pasa al estado cero (lógica positiva), el microprocesador termina la instrucción que está realizando e introduce una instrucción de bifurcación a subrutina en la página cero (ZBSR) en el Registro de Instrucciones. Simultáneamente, pone en estado uno el biestable de Interrupción del Registro de estado y control del programa y responde al sistema externo con las señales INTACK y OPREQ.

Si se produce una petición de interrupción mediante el biestable de Inhibición de Interrupción, que se encuentra en estado uno, será reconocida y atendida la petición tan pronto como dicho biestable vuelva al estado cero.

"Terminal de Operación realizada (OPACK)": La señal de Operación realizada, aplicada a este terminal, es una respuesta procedente de la memoria externa o de los periféricos a una señal de Petición de Operación enviada por el microprocesador. El dispositivo externo (memoria o periférico) indica al procesador que ha realizado la operación, que éste le pidió mediante una conmutación del estado uno al cero de la señal OPACK. Esta forma de operación permite un funcionamiento asíncrono de los periféricos y la memoria con respecto al microprocesador.

"Terminal de detección (SENSE)": El terminal SENSE proporciona un medio de realizar entradas de información al microprocesador, independiente del bus de entrada y salida. El terminal de detección está conectado directamente a uno de los biestables del Registro de estado y control de programa. El estado de este terminal es memorizado cuando el microprocesador realiza una instrucción de cargar el Registro de estado y control del programa (SPSU) o de comprobar dicho estado (TPSU) en el último ciclo de la instrucción correspondiente.

"Terminal de Inhibición de Dirección (ADREN)": Permite controlar el tercer estado de las salidas de dirección ADR0 a ADR12. Cuando la señal ADREN se encuentra en estado lógico uno, las salidas de dirección se llevan al tercer estado, lo cual permite conectarlas a otras señales. Los

terminales ADR13 y ADR14, que están multiplexados con otras señales, no son afectadas por la señal ADREN.

"Terminal de Inhibición del bus de Información (DBUSEN)": Controla el tercer estado del bus de datos. Cuando DBUSEN tiene estado uno, el bus de datos se encuentra en estado flotante y, por lo tanto, puede conectarse a otros dispositivos, formando una "O" por conexión (wired-OR).

"Terminales del bus de datos (DBUS)": Los terminales DB0 a DB7 del bus de datos permiten transferir información desde el microprocesador hacia el exterior o viceversa.

La dirección del flujo de información viene indicada por la señal R/W.

"Terminales del bus de direccionamiento (ADR)": Los terminales ADR0 a ADR14 permiten seleccionar las direcciones de la memoria en las operaciones de escritura y lectura, siendo los bits ADR0 a ADR7 los que seleccionan los periféricos en las operaciones de entrada y salida. La dirección aparece en los terminales citados, mientras la señal OPREQ se encuentra en estado uno y no se necesita, por lo tanto, un registro de dirección externo.

"Terminal de petición de operación (OPREQ)": Este terminal de salida coordina todas las operaciones del microprocesador con el exterior.

Ninguna de las señales presentes en los terminales de salida, cambia de estado, mientras OPRQ se encuentra en un nivel lógico uno, a excepción de WRP. La señal OPRQ permanecerá en dicho nivel, hasta que se termine la operación externa, lo cual vuele indicado por la conmutación de uno a cero de la señal OPACK. El microprocesador no realiza ninguna operación interna, hasta que se produce la citada conmutación de OPACK.

"Terminal de Aceptación de Interrupción (INTACK)": Lo utiliza el microprocesador para responder a las peticiones externas de interrupción.

"Terminal de selección Memoria/Entrada-Salida (M/IO)": Esta es una de las señales, que controla la realización de las operaciones externas. Su estado lógico indica si se va a realizar una transferencia con memoria o con los periféricos.

El estado de la señal M/IO no cambia, mientras la señal OPREQ se encuentra en estado lógico uno.

"Terminal de control de Escritura/Lectura (R/W)": Se trata de una de las señales de control de operaciones externas. Indica si la operación, que se va a realizar, es una Lectura o Escritura de la memoria externa, o una

transferencia de entrada o salida con un periférico. El estado de la señal R/W es válido sólo, cuando la señal OPREQ se encuentra en estado lógico uno, y no cambia mientras esta señal permanece en dicho estado.

"Señal de control de entrada/salida (D/C)": Sirve para diferenciar los dos tipos de instrucciones de entrada/salida de un octeto WRTC, WRTD, REDC, REDD.

El estado de esta señal no debe ser utilizado hasta que la señal OPREQ se encuentre en estado uno lógico y la señal M/IO indique que se va a realizar una transferencia de entrada-salida. La señal D/C utiliza el mismo terminal 18 que el bit 14 de la dirección (ADR14). Cuando la señal M/IO indica que se va a realizar una transferencia de entrada-salida (estado cero), el estado del terminal 18 deberá ser interpretado como D/C. Por el contario, cuando la señal M/IO indica que se va a realizar una transferencia con memoria, el estado de dicho terminal deberá ser interpretado como el bit 14 de la dirección (ADR14).

"Señal de control del tipo de instrucciones de entrada-salida (E/NE)": El estado de la señal de este terminal del microprocesador se utiliza para diferenciar las operaciones de transferencia de entrada-salida de uno y de dos octetos. Por lo tanto, la señal E/NE indica si la información de los 8 bits menos significativos de la dirección (ADR0 a ADR7) deben, o no, ser utilizados en la operación de entrada-salida.

"Terminal de Marcha/Espera (RUN/WAIT)": Indica el estado del microprocesador, así cuando está ejecutando instrucciones (RUN) este terminal está en estado lógico uno y, en caso contrario, (WAIT) en estado cero. El estado de Espera (WAIT) puede producirse por la realización de la instrucción de paro (HALT), o poniendo en estado cero la entrada PAUSE.

"Terminal de Control (FLAG)": Es de salida y determina el estado del bit de control (FLAG) del registro de estado y control de programa.

"Terminal de salida del impulso de escritura (WRP)": La señal de este terminal es un impulso que aparece durante cada operación de escritura en la memoria externa, o de salida de información hacia un periférico.

TRANSFERENCIAS CON PERIFERICOS

El 2650 puede realizar las transferencias de información con los periféricos de diferentes modos, lo cual le permite adaptarse más fácilmente a sus características, que pueden ser muy dispares en lo que se refiere a velocidad de transmisión y forma de enviar la información (en serie, en paralelo, etc.).

Entradas y salidas a través de los terminales de Detección (SENSE) y de Control (FLAG)

El microprocesador puede enviar o recibir del exterior un bit de información, sin necesidad de realizar una decodificación de dirección o utilizar señales de sincronización.

El bit de Control puede ser utilizado con varios fines, de los que son un ejemplo:

a) Salida de información en serie.

b) Bit de dirección adicional para incrementar la capacidad de direccionamiento.

c) Variable de control de sistemas lógicos externos, acoplados al microprocesador.

d) Como generador de impulsos para realizar una selección secuencial de diversos dispositivos periféricos.

El bit de detección (Sense) forma parte del registro de estado y control del programa.

Esta entrada de un bit al microprocesador puede ser utilizada también con diversos fines, entre los cuales se pueden citar:

a) Entrada de información en serie.

b) Entrada de detección del estado de una variable externa.

c) Una señal de ruptura de secuencia de un programa.

Entrada y salida simple (non-extended I/O)

Esta forma de introducir y extraer información del microprocesador 2650 se realiza mediante instrucciones de un sólo octeto. Existen dos instrucciones de entrada simple de información, denominadas REDC y REDD, que se diferencian únicamente en el estado de la señal D/C. El estado de esta señal es un uno lógico, cuando se realiza REDC. De igual forma existen dos instrucciones de salida, WRTC y WRTD, que se distinguen de una forma similar mediante el estado de la señal D/C.

La señal D/C puede ser utilizada como un bit de dirección de periférico en sistemas sencillos.

En la tabla 5-3 se indican los niveles de las señales M/IO, R/W, ADR13-E/NE, ADR14-D/C, durante las diferentes operaciones de entrada-salida y de transferencia con la memoria.

Campo de Clase	Grupo de Instrucciones	Modo de Direccionamiento	Número de octetos	Número de ciclos
0	Aritméticas y de transferencia	Registro	1	2
1	Aritméticas y de transferencia	Inmediato	2	2
2	Aritméticas y de transferencia	Relativo	2	3
3	Aritméticas y de transferencia	Absoluto	3	4
4	Control Entrada/Salida Rotación		1	2
5	Control Entrada/Salida		1-2	3
6	Bifurcación	Relativo	2	3
7	Bifurcación	Absoluto	3	3

Tabla 5-3.—Niveles de diversas señales en diferentes operaciones.

Entrada y salida compuesta (extended I/O)

Existen dos tipos de instrucciones de transferencia de información con los periféricos de dos octetos cada una: una de entrada REDE y otra de salida WRTE. Se denominan compuestas (extended), porque comunican al microprocesador con los periféricos mediante dos octetos. La instrucción REDE hace aparecer el segundo octeto en las líneas de dirección ADR0 a ADR7, que pueden ser utilizadas como líneas de selección de periféricos; el periférico seleccionado deberá presentar la información en el bus de datos (DBUS) y el microprocesador la introducirá en el registro especificado en la instrucción. La instrucción WRTE también hace aparecer en las líneas ADR0 a ADR7 el segundo octeto y, simultáneamente, presenta la información del registro seleccionado en el bus de datos (DBUS), para que sea recogida por el periférico.

Entrada y salida mediante las señales de control de memoria

El usuario del 2650 puede comunicarse con los periféricos mediante las señales de control de la memoria. La ventaja de este método es que se pueden realizar las transferencias de entrada y salida, mediante las ins-

truccciones de transferencia con memoria y aritméticas. Para ello, el diseñador ha de asignar adecuadamente direcciones de memoria a los periféricos para recibir las señales de control de la memoria. Una desventaja de este método es que, en ocasiones, puede ser necesario decodificar más bits de dirección, que cuando se utilizan las instrucciones normales de entrada y salida.

TRATAMIENTO DE LAS INTERRUPCIONES

El sistema de interrupciones del 2650 es de un sólo nivel y direcciones vectorizadas. Existe un terminal de petición de interrupciones, al que los periféricos pueden aplicar una señal de petición de interrupciones (INTREQ), que está constituida por una conmutación del estado lógico uno al cero.

La secuencia de acciones, que se producen al realizarse una interrupción, son las siguientes:

1.ª) Un periférico inicia una interrupción conmutando de uno a cero la señal INTRQ, si el bit de Inhibición de Interrupciones se encuentra en estado lógico cero.

2.ª) El microprocesador termina la instrucción que está realizando, al producirse la interrupción.

3.ª) El microprocesador pone en estado uno el bit de Inhibición de Interrupciones. De esta forma, las interrupciones posteriores quedan inhibidas hasta que este bit es puesto a cero mediante la instrucción adecuada, o se realiza una instrucción de retorno de subrutina y desinhibición de interrupciones.

4.ª) El microprocesador introduce el primer octeto de una instrucción de bifurcación a subrutina en la página cero con el direccionamiento relativo (ZBSR), en el registro de instrucción en lugar de la siguiente instrucción en secuencia.

5.ª) El periférico responde a la señal INTACK enviada por el microprocesador, presentando un octeto en el bus de datos. El microprocesador memoriza dicha información y la utiliza como segundo octeto de la instrucción ZBSR.

6.ª) El microprocesador ejecuta la instrucción de bifurcación a subrutina (ZBSR) (memorizando la dirección de la instrucción del programa que sigue a la última realizada, en la memoria LIFO de direcciones de retorno) y ejecuta la instrucción situada en el octeto cero de la página cero, o la dirección relativa a la página cero suministrada por el periférico.

7.ª) Cuando se termina la subrutina de interrupción, el microprocesador realiza una instrucción de retorno RETC o RETE, introduce la información de la memoria LIFO en el contador de programa, y continúa el programa principal a partir del punto en que se encontraba, cuando se produjo la interrupción.

La figura 5-48 presenta una foto del sistema de desarrollo INSTRUCTOR 50, basado en el microprocesador 2650 de Signetics.

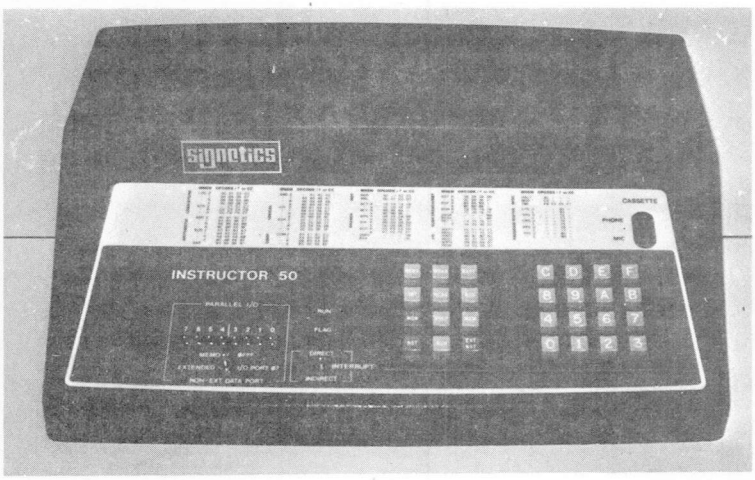


Fig. 5-48.–Sistema de desarrollo INSTRUCTOR 50 para entrenamiento, manejo y evaluación de proyectos, basado en el microprocesador 2650 de Signetics. (Cortesía de ANCART.)

CAPITULO 6

Las instrucciones en los microprocesadores

CARACTERISTICAS Y CONCEPTO DE LAS INSTRUCCIONES

Las instrucciones, que puede ejecutar el microprocesador, las interpreta su Decodificador de instrucciones, cuando se suministran en código máquina, o sea, binario. Como ya se comentó, manejar este código presenta muchos problemas, por lo que, a la hora de confeccionar los programas, se emplean otros lenguajes más inteligibles para el ser humano. Entre estos lenguajes, destaca para su aplicación con los microprocesadores, el llamado lenguaje Ensamblador, basado en los símbolos nemónicos de las instrucciones y que es de carácter elemental, puesto que está muy relacionado con la arquitectura interna de la máquina.

Los símbolos nemónicos del lenguaje Ensamblador se forman, designando a cada instrucción con una o más letras mayúsculas, que son abreviaturas de la palabra, en idioma inglés, significativa de la misión de la instrucción.

En general, además de expresar la instrucción, al confeccionar un programa, es necesario poner a continuación el dato u operando preciso, que interviene en la realización de la instrucción, aunque existen algunas instrucciones que no necesitan operando para poderse realizar. Por ejemplo, la instrucción LDA, Cargar al Acumulador (Load Acumulator), debe ir seguida de otra información supletoria, que indique de alguna forma el dato, que hay que cargar en el Acumulador. Dicha información puede ser directamente el dato en cuestión, o bien, expresar la dirección de la posición de memoria donde se encuentra el mismo.

Respecto a las características del lenguaje Ensamblador, se destina un apartado en este capítulo, para desarrollarlas ampliamente, no obstante, téngase presente que para completar una instrucción en este lenguaje, además del nemónico de la misma y el operando, se reserva un campo previo para la colocación de una *etiqueta*, y un campo final para el *co-*

mentario, que desee añadir el programador para interpretar mejor la finalidad de la instrucción. Una instrucción completa en lenguaje Ensamblador puede ser:

ETIQUETA	NEMONICO	OPERANDO	COMENTARIO
PON	LDA	#27	Cargo 27 en el Acumulador

En el ejemplo anterior la etiqueta es cualquier palabra (PON), que sirve para referenciar esa instrucción y sirve para acudir a ella en cualquier momento, localizando simplemente la etiqueta, evitándose de esta forma tener que llevar la cuenta de las posiciones de memoria, que van ocupando las instrucciones.

Los tipos básicos en que se clasifican las instrucciones, según su misión (aritméticas, de bifurcación, etc.) se describieron en los temas dedicados a los ordenadores en general, y son iguales que las que tratan los microprocesadores.

Las instrucciones de los microprocesadores disponen también de varias formas de direccionamiento, lo que significa, diferentes modos de localizar datos, con los que debe operar la instrucción. A continuación se analizan los modos de direccionamiento más generales, y posteriormente los particulares de cada microprocesador. Como ejemplo de aplicación de una sencilla instrucción, se usará LDA o Carga del Acumulador; también se supondrá que la CPU trabaja con palabras de 8 bits y el bus de direccionamiento es de 16 bits.

Direccionamiento inmediato: Detrás del símbolo nemónico que expresa la instrucción, se proporciona el operando directamente. Una instrucción con este direccionamiento consta de 2 bytes, el 1.º es el código OP de la instrucción, y el 2.º, el operando.

Un ejemplo de instrucción con este tipo de direccionado: LDA 38, que consigue que el Acumulador de 8 bits se cargue con el dato 38 hexadecimal (en binario 0011 1000).

Direccionamiento directo: Detrás del nemónico se coloca la dirección de la posición de la memoria de datos, en donde se encuentra el operando almacenado.

Ejemplo: LDA 53,A1. En la posición 53,A1 se halla el dato, que se ha de cargar en el Acumulador.

Un caso particular de este modo de direccionado es el que recibe el nombre de "por página cero" en el que, detrás del código OP, sólo se coloca el byte que indica la posición de memoria de la página cero (desde la 0 a la 255).

Direccionamiento indirecto: Detrás del nemónico se proporciona la dirección de memoria, cuyo contenido y el de la dirección siguiente forman la dirección de otra posición de memoria, en donde se halla el operando.

Ejemplo: LDA 81,00. En este caso, en la posición 8100 se encuentra el valor 53 y en la posición siguiente 81 01 se encuentra el valor A4. Con ambos valores se forma la nueva dirección de memoria 53,A4 en donde se encuentra el dato a cargar al Acumulador.

Direccionamiento relativo: Detrás del nemónico se da un valor, que, sumado al del Contador de Programa, proporciona la dirección de memoria donde se encuentra el dato.

Ejemplo: LDA 20. El valor 20 (offset) se suma al del Contador de Programa, obteniendo una nueva dirección en donde se encuentra el dato, que acompaña a la instrucción.

Direccionamiento indexado: Detrás del nemónico se coloca un valor, que, sumado al contenido de un registro Indice, facilita la dirección de memoria en donde se halla el operando.

Ejemplo: LDA 18. El valor 18 se suma al valor contenido en el registro Indice, obteniéndose una dirección de memoria, que contiene el valor, que se ha de cargar en el Acumulador.

Direccionamiento por registro de dirección: La CPU dispone de uno o más registros, desde los que se direcciona la memoria.

La potencia del software de un microprocesador depende del número y características de sus instrucciones, y entre éstas últimas destaca los diversos modos de direccionamiento que posean.

Si una instrucción cualquiera de un microprocesador admite diversos modos de direccionamiento, habrá para cada uno de ellos un código OP diferente, para que los datos, que siguen al código de la instrucción, sean tratados de acuerdo con el direccionado que se utiliza.

TRATAMIENTO GENERAL DE LAS INSTRUCCIONES

Se desea realizar una instrucción que cargue al Acumulador con el contenido de la posición de memoria 5E,44, es decir, se trata de una instrucción con direccionamiento directo. La instrucción completa ocupará 3 bytes ó 3 posicicones de la memoria; una inicial, que contiene el código OP de la instrucción; y las dos restantes, la dirección de la memoria que contiene el operando. La representación de los contenidos de la memoria se expone en la figura 6-1 y en ella se aprecia que se ha atribuido al código OP de la instrucción LDA el valor AD_{16} para direccionado directo.

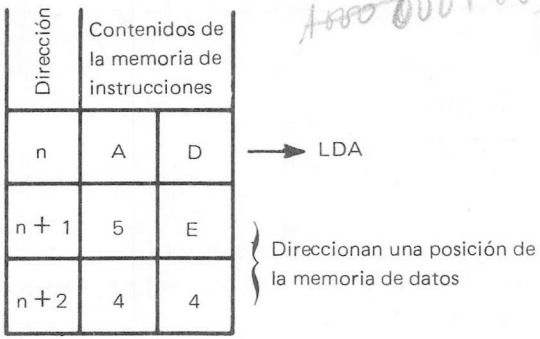


Fig. 6-1.–Disposición de la memoria para contener una instrucción LDA con direccionamiento directo.

Cuando el Contador de Programa recibe la posición n, la lleva por el bus de direccionamiento a la memoria, que decodifica y selecciona dicha posición n. El contenido de la posición n (AD), según la figura 6-1, se transfiere por el bus de instrucciones y datos hasta el Decodificador de instrucciones, el cual interpreta su código y confirma que se trata de una instrucción de carga del Acumulador direccionada directamente. Con ello la Unidad de Control genera las señales adecuadas para buscar el contenido de la posición de la memoria, definida por las 2 siguientes posiciones de la memoria de instrucciones, es decir, la n + 1 y la n + 2.

El PC se incrementa y pasa a contener y direccionar a n + 1, con lo que saca el dato que hay en esa posición (5E) y lo traslada al byte de más peso del Registro de direcciones de la memoria de datos, tal como se ha representado en la figura 6-2.

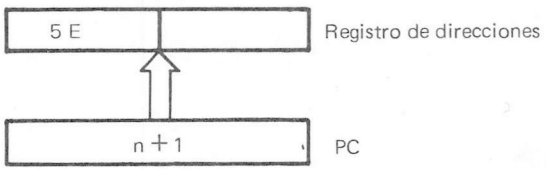


Fig. 6-2.–Transferencia del contenido de la dirección n + 1 al Registro de direcciones.

A continuación, el PC se vuelve a incrementar y pasa a valer n + 2, transfiriendo el contenido de esta posición de la memoria de instrucciones hasta el byte de menos peso del Registro de direcciones de la memoria. (Figura 6-3.)

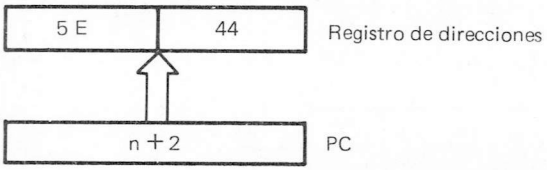


Fig. 6-3.–El Registro de direcciones queda completo al recibir el byte de menos peso, contenido en la posición n + 2.

Una vez completado el Registro de direcciones de la memoria de datos con el valor 5E,44, se pasa a seleccionar esta dirección, y su contenido es cargado con el Acumulador.

La instrucción, que se acaba de analizar, consta de 3 bytes y ocupa 3 posiciones de la memoria de instrucciones. Otras instrucciones sólo ocupan 2 bytes, como sucede con la misma instrucción LDA, pero con direccionado inmediato. Finalmente, hay instrucciónes de un solo byte, puesto que quedan definidas por sí mismas; un ejemplo de este tipo de instrucciones puede ser la que pone a 1 el flag de acarreo o carry.

A continuación y como en el capítulo anterior, se pasa a exponer el juego de instrucciones y las características de los modelos de microprocesadores que se describen.

1.ª PARTE: JUEGO DE INSTRUCIONES DE LAS CPU, R 6500

MODOS DE DIRECCIONAMIENTO

Los microprocesadores de la familia R 6500, fabricados por MOS TECHNOLOGY y por ROCKWELL, disponen de un juego de 56 instrucciones, pudiendo admitir éstas hasta un total de 13 modos de direccionado diferentes. Hay instrucciones que sólo disponen de un modo de direccionado, mientras que otras más potentes tienen varios. Los 13 modos de direccionamiento son los siguientes:

1.º) *Direccionamiento inmediato:* El dato u operando, que trata la instrucción, se coloca "inmediatamente" detrás del código OP de la misma. Las instrucciones con este direccionado serán de 2 bytes.

2.º) *Direccionamiento por página cero:* El operando, que trata la instrucción, está contenido en una posición de la página cero de la memoria. La estructura paginada de la memoria significa que, de los 16 bits del bus de direccionamiento, los 8 de más peso direccionan la página (un total de 256 páginas) y los 8 bits de menos peso determinan la posición dentro de la página (256 posiciones en cada página). La composición del direccionamiento, en general, se muestra en la figura 6-4.

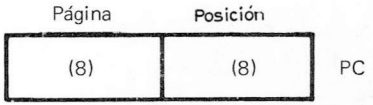


Fig. 6-4.–Función de los 2 bytes para direccionamiento de la memoria.

Con este modo de direccionado el 1.º byte de direccionamiento de memoria es 00 y sólo se precisa fijar con el 2.º byte la posición dentro de la página cero. La instrucción con direccionamiento por página cero dispondrá de 2 bytes en la memoria de instrucciones: el 1.º, que será el código OP y el 2.º la posición de la página cero, en donde se encuentra almacenado el operando

3.º) *Direccionado por página cero, Indice X:* Direcciona una posición de la página cero, como resultado de sumar el 2.º byte que forma la instrucción (el 1.º contiene el código OP) al valor almacenado en el registro Indice X.

Las instrucciones con este tipo de direccionado constan de 2 bytes y permiten que una instrucción fija se convierta en variable, al formar el direccionado con la suma del contenido del registro Indice X.

4.°) *Direccionamiento absoluto:* A este tipo de direccionamiento se le llamó anteriormente "directo" y, con él, la instrucción dispone del operando que trata en la posición de memoria fijada por 2 bytes, que siguen al del código OP. Este tipo de instrucciones ocuparán 3 posiciones en la memoria del programa.

5.°) *Direccionado absoluto, Indice X:* Con este direccionado se suma a la posición indicada en la instrucción el contenido del registro Indice X (por ser de 8 bits, lo máximo que puede valer es 256). Se trata de instrucciones de 3 bytes.

6.°) *Direccionado absoluto, Indice Y:* Igual que el tipo anterior, pero sumando el contenido del registro Indice Y.

7.°) *Direccionado indexado, indirecto, X:* El 2.° byte de la instrucción se suma al contenido del registro Indice X. El resultado de esta suma apunta a una posición de la página cero, que contiene los 8 bits de menos peso de la dirección efectiva donde se encuentra el dato. La siguiente posición de la página cero contiene los 8 bits de más peso de la dirección efectiva.

Este tipo de direccionado representado en la figura 6-5, se usa para acceder a listas de datos.

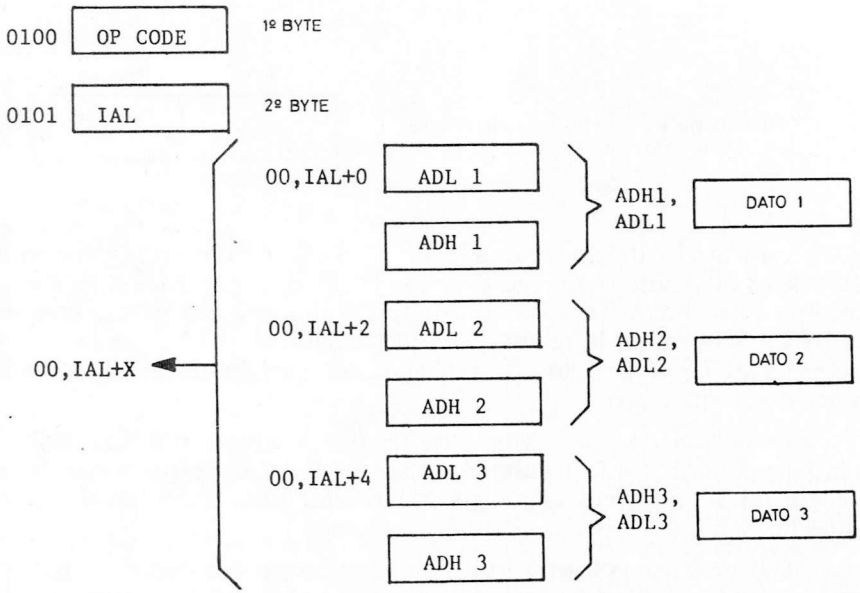


Fig. 6-5.–Direccionado indexado indirecto X.

8.°) *Direccionado indirecto, indexado, Y:* El 2.° byte de esta instrucción apunta a una posición de la página cero, a cuyo contenido se suma el del registro Indice Y, siendo el resultado de la suma los 8 bits de menos peso de la dirección efectiva de la memoria, donde se encuentra el operando. El carry de esta suma, si lo hay, se suma al contenido de la siguiente posición de la página cero y así se consiguen los 8 bits de más peso de la dirección efectiva. En la figura 6-6 se presenta gráficamente este tipo de direccionamiento.

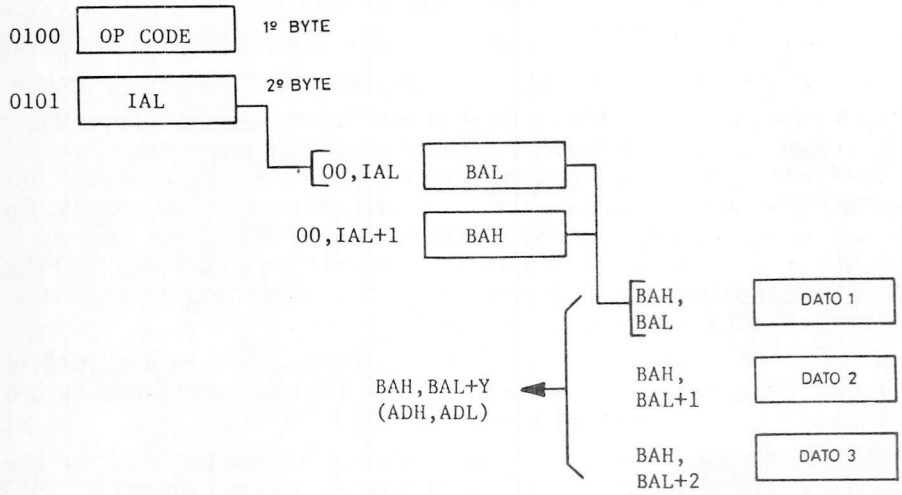


Fig. 6-6.—Direccionado indirecto, indexado, Y.

9.°) *Direccionado por Acumulador:* Responde a una instrucción de un solo byte, que implica una operación en la que participa el Acumulador.

10.°) *Direccionado por página cero, Indice Y:* Igual que el direccionado por página cero Indice X, pero ahora interviniendo el Indice Y.

11.°) *Direccionado relativo:* Se usa sólo en las instrucciones de bifurcación o branch. El 2.° byte es un "offset", que hay que sumar al contenido del Contador de Programa.

12.°) *Direccionado absoluto indirecto:* Se usa exclusivamente en la instrucción de salto JMP, en la que el 2.° y 3.° byte indican los 2 bytes que forman la nueva dirección a saltar en el programa y que se carga en el Contador de Programa.

REPERTORIO DE INSTRUCCIONES DE LOS MICROPROCESADORES R 6500

El conjunto de instrucciones que dispone cualquier CPU debe comprender algunos tipos, que son fundamentales para el procesamiento de los datos. Se indican a continuación los más importantes:

a) Instrucciones que permitan la realización de operaciones lógicas y aritméticas.

b) Instrucciones para leer y escribir en la memoria.

c) Instrucciones de intercambio de contenidos entre los registros.

d) Instrucciones para efectuar roturas de secuencia en el programa.

e) Instrucciones de salto a subrutina y retorno, que trasladan al PC a posiciones de la memoria en donde se guardan programas de operaciones repetitivas del programa principal. Una vez efectuados dichos programas o subrutinas, el PC retorna a la posición de partida. En estas instrucciones interviene el puntero de Stack, que salva en la parte de memoria que controla, tipo LIFO, los contenidos del PC y otros registros importantes de la posición de partida, para permitir el retorno a la misma.

Las CPU de la familia R 6500 poseen 56 instrucciones básicas, pero, al permitir varios modos de direccionamiento (13), se convierten en 160 combinaciones de códigos OP diferentes.

A continuación, se explican brevemente cada una de las 56 instrucciones, por orden alfabético y, posteriormente, se presenta un cuadro resumido de dichas instrucciones con sus nemónicos, códigos OP, ciclos y bytes que ocupan.

1.ª) *ADC:* Suma el contenido de la posición de memoria direccionada, representado por (M) con el contenido del Acumulador y del acarreo o carry (C), depositando el resultado en el Acumulador y en el carry.

$$A + (M) + C \rightarrow A, C$$

El flag N queda con el mismo estado que el bit correspondiente del Acumulador. Si, como resultado de la operación todos los bits del Acumulador quedan a cero, el flag Z pasa a valer 1. Los flags afectados son: N, V, C y Z.

2.ª) *AND:* Realiza la operación lógica AND entre los 8 bits de la posición de memoria direccionada y los 8 bits del Acumulador. Flags afectados: N y Z

$$A \leftarrow A \wedge (M)$$

Ejemplo:

```
    1000 1111
    0111 1100
  ------------
AND: 0000 1100
```

3.ª) *ASL:* Pone un cero en el bit de menos peso, bien del Acumulador o bien del contenido de la posición de memoria direccionada y corre todos los bits una posición a la izquierda, pasando el bit de más peso al carry, tal como se representa en la figura 6-7. Flags a los que afecta: N, Z y C.

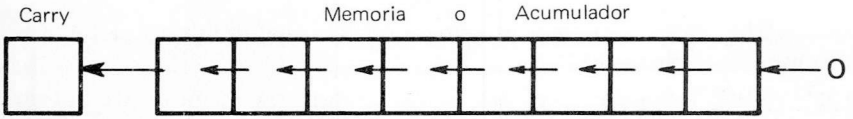


Fig. 6-7.–Significado gráfico de la instrucción ASL.

4.ª) *BCC:* Bifurcación o branch si el carry vale cero. Se trata de una instrucción de 2 bytes y tiene un direccionamiento relativo respecto a la posición que ocupa ella misma. El 2.º byte indica el número de posiciones hacia adelante o hacia atrás (en este caso el valor se pone en forma de complemento a 2) que ha de saltar al PC.

5.ª) *BCS:* Branch similar al de la anterior instrucción, pero ahora se realiza si el carry vale 1 (set).

6.ª) *BEQ:* Branch si Z = 1, es decir, se produce la bifurcación cuando en la operación anterior el Acumulador ha quedado con todos sus bits a cero.

7.ª) *BIT:* Realiza una operación AND entre el contenido del Acumulador y el de la posición de la memoria direccionada. El resultado de la operación no se deposita en el Acumulador, como es costumbre, si no que se pierde y queda representado sólo por los flags N, Z y V del registr de estado P.

8.ª) *BMI:* Branch si el resultado de la operación anterior es negativo, o sea, si el flag N = 1.

9.ª) *BNE:* Branch cuando el flag Z = 0.

10.ª) *BPL:* Branch si el flag N = 0, o sea, si el resultado de la operación anterior ha sido positivo.

11.ª) *BRK:* Esta instrucción origina una secuencia de interrupción (IRQ), almacenando en el stack el PC y el registro de estado P. El PC pasa a contener el valor del vector interrupción asignado a las posiciones de memoria FFFE y FFFF, tal como se expresa gráficamente:

$$PC + 2\downarrow \quad P\downarrow$$
$$(FFFE) \rightarrow PCL$$
$$(FFFF) \rightarrow PCH \quad \text{Afecta al flag B.}$$

12.ª) *BVC:* Branch, si el flag V = 0, o sea, cuando no hay sobrepasamiento en el 7.º bit.

13.ª) *BVS:* Branch, si el flag V = 1, o sea, si se ha producido sobrepasamiento.

14.ª) *CLC:* Esta instrucción pone a cero el flag del carry. Tiene un direccionamiento implicado.

15.ª) *CLD:* Pone a cero el flag D, que determina el modo de operar en forma decimal en todas las operaciones aritméticas.

16.ª) *CLI:* Pone a cero el flag I de las interrupciones.

17.ª) *CLV:* Pone a cero el flag V.

18.ª) *CMP:* Compara el contenido de la posición de memoria direccionada con el del Acumulador, realizando la operación A – M y no depositando el resultado de la misma en ningún lado. El resultado sólo afecta a los flags N, Z y C.

19.ª) *CPX:* Compara el contenido de la posición de memoria direccionada con el del registro Indice X, realizando la operación X – M, de forma similar a la instrucción anterior. Afecta a N, Z y C.

20.ª) *CPY:* Igual que la anterior pero manejando el registro Indice Y.

21.ª) *DEC:* Decrementa una unidad el contenido de la memoria especificada. Afecta a N y Z.

$$(M) - 1 \rightarrow M$$

22.ª) *DEX:* Decrementa el contenido del registro Indice X.

$$(X) - 1 \rightarrow X$$

23.ª) *DEY:* Decrementa el contenido de Y.

$$(Y) - 1 \rightarrow Y$$

24.ª) *EOR:* Efectúa la operación O-Exclusiva entre el contenido de la posición de memoria direccionada y el Acumulador. Afecta a N y Z.

$$A \forall M \rightarrow A$$

25.ª) *INC:* Incrementa una unidad el contenido de la posición de memoria direccionada. Afecta a N y Z.

$$(M) + 1 \rightarrow M$$

26.ª) *INX:* Incrementa 1 al contenido de X. $(X) + 1 \rightarrow X$.

27.ª) *INY:* Incrementa en 1 al contenido del registro Y . $(Y) + 1 \rightarrow Y$.

28.ª) *JMP:* Salto o rotura de secuencia incondicional a la dirección que determina la misma instrucción.

Se trata de una instrucción de 3 bytes, dedicándose el 1.º al código OP y los dos restantes a indicar la posición de la moria de instrucciones a la que se produce el salto. Este tipo de salto es el absoluto. Existe también el salto indirecto, con el cual se puede dirigir el salto a una nueva instrucción, según el número de veces que se sale del bucle en el que se produce, tal como se indica en la figura 6-8.

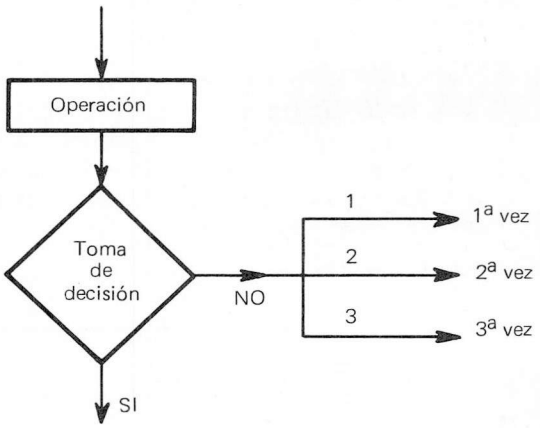


Fig. 6-8.—En el salto indirecto, el PC se dirige a una nueva instrucción según el número de veces que se haya efectuado el bucle.

Así en la figura 6-8, si se sale del bucle la 1.ª vez, el PC se dirige a la instrucción que determina la línea 1, si se sale del bucle la 2.ª vez, a la que determina la línea 2 y así sucesivamente.

29.ª) *JSR:* Salto a subrutina, guardándose el estado de la posición de retorno del programa principal mediante el stack.

La instrucción JMP traslada el PC a una nueva instrucción, sin tener en cuenta de la que se parte. Con JSR se almacena el estado de la posición de partida, a la que se retorna una vez completada la subrutina, la cual consiste en un pequeño programa, que se repite varias veces dentro del programa principal.

$$PC + 2\downarrow (PC + 1) \rightarrow PCL$$
$$(PC + 2) \rightarrow PCH$$

30.ª) *LDA:* Carga en el Acumulador el contenido de la posición de memoria direccionada. Afecta a N y Z.

$$(M) \rightarrow A$$

31.ª) *LDX:* Carga el contenido de la posición de la memoria direccionada, en el registro Indice X. Afecta a los flags N y Z.

$$(M) \rightarrow X$$

32.ª) *LDY:* Igual que la instrucción anterior, pero afectando al registro Y.

$$(M) \rightarrow Y$$

33.ª) *LSR:* Se trata de una instrucción de desplazamiento, que puede afectar al Acumulador o a una posición direccionada de la memoria. Desplaza a todos los bits una posición hacia la derecha, colocando un 0 en el de más peso y trasladando el de menos peso al flag carry, como queda representado en la figura 6-9.

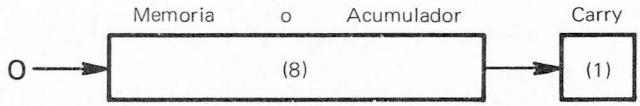


Fig. 6-9.–Representación gráfica de la instrucción LSR. Afecta a N, Z y C.

34.ª) *NOP:* Consume inútilmente 2 ciclos de la máquina y tiene una longitud de 1 byte. No hace ninguna operación. El microprocesador la lee y pasa a la siguiente instrucción, perdiendo tiempo, únicamente.

35.ª) *ORA:* Realiza la operación lógica OR entre el Acumulador y la posición de memoria direccionada. Afecta a N y Z.

$$A \vee (M) \rightarrow A$$

36.ª) *PHA:* Es una instrucción que no afecta a los flags. Guarda los 8 bits de contenido del Acumulador en la posición de memoria direccionada por el stack, incrementando en 1 al puntero de stack, que, de esta forma, queda apuntando la próxima posición. Gráficamente se expresa A↓.

37.ª) *PHP:* Guarda el contenido del registro de estado P en la memoria a través del direccionado del puntero del stack. P↓.

38.ª) *PLP:* Se recupera el registro de estado P de las posiciones gobernadas por el stack. P↑. Afecta a todos los flags.

39.ª) *PLA:* Se recupera el registro Acumulador de la posición gobernada por el stack. Afecta a N y Z.

40.ª) *ROL:* Rotación de la izquierda del Acumulador o de la posición de memoria direccionada. Afecta a N, Z y C.

41.ª) *ROR:* Rotación a la derecha de los bits del Acumulador o de la posición de memoria direccionada, participando el carry, como se representa en la figura 6-10.

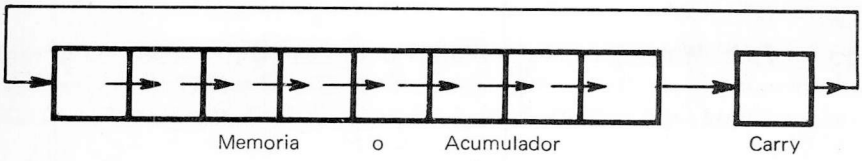


Fig. 6-10.–Representación gráfica de la instrucción ROR. Afecta a N, Z y C.

42.ª) *RTI:* Retorno desde una interrupción. Regenera el PC y el P. Afecta a todos los flags.

$$PC\uparrow\ P\uparrow$$

43.ª) *RTS:* Retorno desde subrutina. Regenera el PC.

$$PC\uparrow : PC + 1 \rightarrow PC$$

44.ª) *SBC:* Resta aritmética entre el Acumulador y la posición de memoria direccionada y la llevada (prestado) C complementada. Afecta a los flags N, Z, C y V.

$$A - (M) - \overline{C} \rightarrow A$$

C = 1, si resultado < 0
V = 1, si: −128 < resultado > + 127
N = 1, si bit 7 de resultado es 1.

45.ª) *SEC:* Pone a 1 o set al flag carry. 1 → C.

46.ª) *SED:* Pone a 1 el flag D, que afecta al modo de operar (decimal).

47.ª) *SEI:* Pone a 1 al flag I, que evita las interrupciones mascarables.

48.ª) *STA:* Instrucción opuesta a LDA. A → (M).

49.ª) *STX:* X → (M).

50.ª) *STY:* Y → (M).

51.ª) *TAX:* Transfiere el contenido del Acumulador al registro X. Afecta a los flags N y Z. A → X.

52.ª) *TAY:* A → Y.

53.ª) *TYA:* Transfiere el contenido del registro Y al Acumulador.

$$Y \rightarrow A$$

54.ª) *TXA:* X → A.

55.ª) *TSX:* Transfiere el contenido del puntero del stack, S, al registro Indice X.

$$S \rightarrow X$$

56.ª) *TXS:* Transfiere el contenido del registro Indice X al puntero de stack, S.

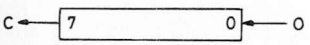
$$X \rightarrow S$$

CUADRO RESUMIDO DEL JUEGO DE INSTRUCCIONES DE LAS CPU R 6500

A continuación, se presenta un resumen de las características más destacables del juego de instrucciones de los microprocesadores de la familia R 6500. Cada instrucción viene definida por las diversas formas de direc-

cionamiento que admite, junto con su código OP correspondiente, expresado en hexadecimal. También se indica el número de bytes de que consta cada direccionado de cada instrucción, así como el número de ciclos máquina que tarda en ejecutarse.

JUEGO DE INSTRUCCIONES DE LAS CPU R 6500

Nemónico y función	Direccionamiento	Código OP	Número bytes	Número ciclos
1.ª) ADC A + (M) + C → A, C	Inmediato	69	2	2
	Página cero	65	2	3
	Página cero, X	75	2	4
	Absoluto	6D	3	4
	Absoluto, X	7D	3	4
	Absoluto, Y	79	3	4
	Indirecto, X	61	2	6
	Indirecto, Y	71	2	5
2.ª) AND A ∧ (M) → A	Inmediato	29	2	2
	Página cero	25	2	3
	Página cero, X	35	2	4
	Absoluto	2D	3	4
	Absoluto X	3D	3	4*
	Absoluto, Y	39	3	4*
	Indirecto, X	21	2	6
	Indirecto, Y	31	2	5*
3.ª) ASL 	Acumulador	0A	1	2
	Página cero	06	2	5
	Página cero, X	16	2	6
	Absoluto	0E	3	6
	Absoluto, X	1E	3	7
4.ª) BCC Branch, si C = 0	Relativo	90	2	2*
5.ª) BCS Branch, si C = 1	Relativo	B0	2	2*
6.ª) BEQ Branch, si Z = 1	Relativo	F0	2	2*
7.ª) BIT A ∧ (M)	Página cero	24	2	3
	Absoluto	2C	3	4
8.ª) BMI Branch, si N = 1	Relativo	30	2	2*

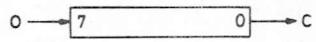
JUEGO DE INSTRUCCIONES DE LAS CPU R 6500 (Continuación)

Nemónico y función	Direccionamiento	Código OP	Número bytes	Número ciclos
9.ª) BNE Branch, si Z = 0	Relativo	D0	2	2*
10.ª) BPL Branch, si N = 0	Relativo	10	2	2*
11.ª) BRK Ver explicación	Implicado	00	1	7
12.ª) BVC Branch, si V = 0	Relativo	50	2	2*
13.ª) BVS Branch, si V = 1	Relativo	70	2	2*
14.ª) CLC C ← 0	Implicado	18	1	2
15.ª) CLD D ← 0	Implicado	D8	1	2
16.ª) CLI I ← 0	Implicado	58	1	2
17.ª) CLV V ← 0	Implicado	B8	1	2
18.ª) CMP A – (M)	Inmediato	C9	2	2
	Página cero	C5	2	3
	Página cero, X	D5	2	4
	Absoluto	CD	3	4
	Absoluto, X	DD	3	4*
	Absoluto, Y	D9	3	4*
	Indirecto, X	C1	2	6
	Indirecto, Y	D1	2	5*
19.ª) CPX X – (M)	Inmediato	E0	2	2
	Página cero	E4	2	3
	Absoluto	EC	3	4
20.ª) CPY Y – (M)	Inmediato	C0	2	2
	Página cero	C4	2	3
	Absoluto	CC	3	4

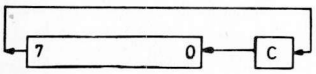
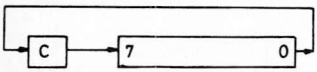
JUEGO DE INSTRUCCIONES DE LAS CPU R 6500 (Continuación)

Nemónico y función	Direccionamiento	Código OP	Número bytes	Número ciclos
21.ª) DEC (M) − 1 → M	Página cero	C6	2	5
	Página cero, X	D6	2	6
	Absoluto	CE	3	6
	Absoluto, X	DE	3	7
22.ª) DEX (X) − 1 → X	Implicado	CA	1	2
23.ª) DEY (Y) − 1 → Y	Implicado	88	1	2
24.ª) EOR A ∀ (M) → A	Inmediato	49	2	2
	Página cero	45	2	3
	Página cero, X	55	2	4
	Absoluto	4D	3	4
	Absoluto, X	5D	3	4*
	Absoluto, Y	59	3	4*
	Indirecto, X	41	2	6
	Indirecto, Y	51	2	5*
25.ª) INC (M) + 1 → M	Página cero	E6	2	5
	Página cero, X	F6	2	6
	Absoluto	EE	3	6
	Absoluto, X	FE	3	7
26.ª) INX (X) + 1 → X	Implicado	E8	1	2
27.ª) INY (Y) + 1 → Y	Implicado	C8	1	2
28.ª) JMP Salto a una nueva dirección	Absoluto	4C	3	3
	Indirecto	6C	3	5
29.ª) JSR Salto a subrutina	Absoluto	20	3	6
30.ª) LDA (M) → A	Inmediato	A9	2	2
	Página cero	A5	2	3
	Página cero, X	B5	2	4
	Absoluto	AD	3	4
	Absoluto, X	BD	3	4*
	Absoluto, Y	B9	3	4*

JUEGO DE INSTRUCCIONES DE LAS CPU R 6500 (Continuación)

Nemónico y función	Direccionamiento	Código OP	Número bytes	Número ciclos
	Indirecto, X	A1	2	6
	Indirecto, Y	B1	2	5*
31.[a]) LDX (M) → X	Inmediato	A2	2	2
	Página cero	A6	2	3
	Página cero, Y	B6	2	4
	Absoluto	AE	3	4
	Absoluto, Y	BE	3	4*
32.[a]) LDY Y ← (M)	Inmediato	A0	2	2
	Página cero	A4	2	3
	Página cero, X	B4	2	4
	Absoluto	AC	3	4
	Absoluto, X	BC	3	4*
33.[a]) LSR  	Página cero	46	2	5
	Acumulador	4A	1	2
	Página cero, X	56	2	6
	Absoluto	4E	3	6
	Absoluto, X	5E	3	7
34.[a]) NOP No hace nada	Implicado	EA	1	2
35.[a]) ORA A ∧ (M) → A	Inmediato	09	2	2
	Página cero	05	2	3
	Página cero, X	15	2	4
	Absoluto	0D	3	4
	Absoluto, X	1D	3	4*
	Absoluto, Y	19	3	4*
	Indirecto, X	01	2	6
	Indirecto, Y	11	2	5*
36.[a]) PHA A → (M_S); S − 1 → S	Implicado	48	1	3
37.[a]) PHP P → (M_S); S − 1 → S	Implicado	08	1	3
38.[a]) PLA S + 1 → S; (M_S) → A	Implicado	68	1	4
39.[a]) PLP S + 1 → S; (M_S) → P	Implicado	28	1	4

JUEGO DE INSTRUCCIONES DE LAS CPU R 6500 (Continuación)

Nemónico y función	Direccionamiento	Código OP	Número bytes	Número ciclos
40.ª) ROL 	Acumulador	2A	1	2
	Página cero	26	2	5
	Página cero, X	36	2	6
	Absoluto	2E	3	6
	Absoluto, X	3E	3	7
41.ª) ROR 	Acumulador	6A	1	2
	Página cero	66	2	5
	Página cero, X	76	2	6
	Absoluto	6E	3	6
	Absoluto, X	7E	3	7
42.ª) RTI Retorno interrup.	Implicado	40	1	6
43.ª) RTS Retorno subrutina	Implicado	60	1	6
44.ª) SBC $A - (M) - \overline{C} \rightarrow A$	Inmediato	E9	2	2
	Página cero	E5	2	3
	Página cero, X	F5	2	4
	Absoluto	ED	3	4
	Absoluto, X	FD	3	4*
	Absoluto, Y	F9	3	4*
	Indirecto, X	E1	2	6
	Indirecto, Y	F1	2	5*
45.ª) SEC $1 \rightarrow C$	Implicado	38	1	2
46.ª) SED $D \rightarrow 1$	Implicado	F8	1	2
47.ª) SEI $1 \rightarrow I$	Implicado	78	1	2
48.ª) STA	Página cero	85	2	3
	Página cero, X	95	2	4
	Absoluto	8D	3	4
	Absoluto, X	9D	3	5
	Absoluto, Y	99	3	5
	Indirecto, X	81	2	6
	Indirecto, Y	91	2	6

JUEGO DE INSTRUCCIONES DE LAS CPU R 6500 (Continuación)

Nemónicc y función	Direccionamiento	Código OP	Número bytes	Número ciclos
49.ª) STX X → (M)	Página cero Página cero, Y Absoluto	86 96 8E	2 2 3	3 4 4
50.ª) STY Y → (M)	Página cero Página cero, X Absoluto	84 94 8C	2 2 3	3 4 4
51.ª) TAX A → X	Implicado	AA	1	2
52.ª) TAX A → Y	Implicado	A8	1	2
53.ª) TYA Y → A	Implicado	98	1	2
54.ª) TSX S → X	Implicado	BA	1	2
55.ª) TXA X → A	Implicado	8A	1	2
56.ª) TXS X → S	Implicado	9A	1	2

Nota: Aquellas formas de direccionamiento de las instrucciones, en las que se ha colocado un asterisco al lado del número de ciclos máquina que poseen, se les debe añadir un ciclo más en el caso de traspasar el límite de la página. Si las instrucciones son de branch, se añade un ciclo cuando el branch se produce en la misma página y 2 si el branch sucede en otra página diferente de la de partida.

A continuación, se presenta un cuadro resumido de las instrucciones de los microprocesadores de la familia R 6500, ofrecido por su fabricante.

INSTRUCTIONS		IMMEDIATE			ABSOLUTE			ZERO PAGE			ACCUM			IMPLIED			(IND, X)			(IND), Y			Z. PAGE, X			ABS, X			ABS, Y			RELATIVE			INDIRECT			Z. PAGE, Y			PROCESSOR STATUS CODES	
MNEMONIC	OPERATION	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	OP	n	#	7 6 5 4 3 2 1 0 N V · B D I Z C	MNEMONIC
ADC	A + M + C → A (4) (1)	69	2	2	6D	4	3	65	3	2							61	6	2	71	5	2	75	4	2	7D	4	3	79	4	3										N V · · · · Z C	ADC
AND	A∧M → A (1)	29	2	2	2D	4	3	25	3	2							21	6	2	31	5	2	35	4	2	3D	4	3	39	4	3										N · · · · · Z ·	AND
ASL	C ← [7 6 5 4 3 2 1 0] ← 0				0E	6	3	06	5	2	0A	2	1										16	6	2	1E	7	3													N · · · · · Z C	ASL
BCC	BRANCH ON C = 0 (2)																															90	2	2							· · · · · · · ·	BCC
BCS	BRANCH ON C = 1 (2)																															B0	2	2							· · · · · · · ·	BCS
BEQ	BRANCH ON Z = 1 (2)																															F0	2	2							· · · · · · · ·	BEQ
BIT	A∧M				2C	4	3	24	3	2																															M_7 M_6 · · · · Z ·	BIT
BMI	BRANCH ON N = 1 (2)																															30	2	2							· · · · · · · ·	BMI
BNE	BRANCH ON Z = 0 (2)																															D0	2	2							· · · · · · · ·	BNE
BPL	BRANCH ON N = 0 (2)																															10	2	2							· · · · · · · ·	BPL
BRK	BREAK (See Fig. 1)													00	7	1																									· · · 1 · 1 · ·	BRK
BVC	BRANCH ON V = 0 (2)																															50	2	2							· · · · · · · ·	BVC
BVS	BRANCH ON V = 1 (2)																															70	2	2							· · · · · · · ·	BVS
CLC	0 → C													18	2	1																									· · · · · · · 0	CLC
CLD	0 → D													D8	2	1																									· · · · 0 · · ·	CLD
CLI	0 → I													58	2	1																									· · · · · 0 · ·	CLI
CLV	0 → V													B8	2	1																									· 0 · · · · · ·	CLV
CMP	A − M	C9	2	2	CD	4	3	C5	3	2							C1	6	2	D1	5	2	D5	4	2	DD	4	3	D9	4	3										N · · · · · Z C	CMP
CPX	X − M	E0	2	2	EC	4	3	E4	3	2																															N · · · · · Z C	CPX
CPY	Y − M	C0	2	2	CC	4	3	C4	3	2																															N · · · · · Z C	CPY
DEC	M − 1 → M				CE	6	3	C6	5	2													D6	6	2	DE	7	3													N · · · · · Z ·	DEC
DEX	X − 1 → X													CA	2	1																									N · · · · · Z ·	DEX
DEY	Y − 1 → Y													88	2	1																									N · · · · · Z ·	DEY
EOR	A ∨ M → A (1)	49	2	2	4D	4	3	45	3	2							41	6	2	51	5	2	55	4	2	5D	4	3	59	4	3										N · · · · · Z ·	EOR
INC	M + 1 → M				EE	6	3	E6	5	2													F6	6	2	FE	7	3													N · · · · · Z ·	INC
INX	X + 1 → X													E8	2	1																									N · · · · · Z ·	INX
INY	Y + 1 → Y													C8	2	1																									N · · · · · Z ·	INY
JMP	JUMP TO NEW LOC				4C	3	3																												6C	5	3				· · · · · · · ·	JMP
JSR	JUMP SUB (See Fig. 2)				20	6	3																																		· · · · · · · ·	JSR
LDA	M → A (1)	A9	2	2	AD	4	3	A5	3	2							A1	6	2	B1	5	2	B5	4	2	BD	4	3	B9	4	3										N · · · · · Z ·	LDA
LDX	M → X (1)	A2	2	2	AE	4	3	A6	3	2																			BE	4	3							B6	4	2	N · · · · · Z ·	LDX
LDY	M → Y (1)	A0	2	2	AC	4	3	A4	3	2													B4	4	2	BC	4	3													N · · · · · Z ·	LDY
LSR	0 → [7 6 5 4 3 2 1 0] → C				4E	6	3	46	5	2	4A	2	1										56	6	2	5E	7	3													0 · · · · · Z C	LSR
NOP	NO OPERATION													EA	2	1																									· · · · · · · ·	NOP
ORA	A ∨ M → A	09	2	2	0D	4	3	05	3	2							01	6	2	11	5	2	15	4	2	1D	4	3	19	4	3										N · · · · · Z ·	ORA
PHA	A → Ms S − 1 → S													48	3	1																									· · · · · · · ·	PHA
PHP	P → Ms S − 1 → S													08	3	1																									· · · · · · · ·	PHP
PLA	S + 1 → S Ms → A													68	4	1																									N · · · · · Z ·	PLA
PLP	S + 1 → S Ms → P													28	4	1																									(RESTORED)	PLP
ROL	[7 6 5 4 3 2 1 0] ← C				2E	6	3	26	5	2	2A	2	1										36	6	2	3E	7	3													N · · · · · Z C	ROL
ROR	C → [7 6 5 4 3 2 1 0]				6E	6	3	66	5	2	6A	2	1										76	6	2	7E	7	3													N · · · · · Z C	ROR
RTI	RTRN INT (See Fig. 1)													40	6	1																									(RESTORED)	RTI
RTS	RTRN SUB (See Fig. 2)													60	6	1																									· · · · · · · ·	RTS
SBC	A − M − C → A (1)	E9	2	2	ED	4	3	E5	3	2							E1	6	2	F1	5	2	F5	4	2	FD	4	3	F9	4	3										N V · · · · Z (3)	SBC
SEC	1 → C													38	2	1																									· · · · · · · 1	SEC
SED	1 → D													F8	2	1																									· · · · 1 · · ·	SED
SEI	1 → I													78	2	1																									· · · · · 1 · ·	SEI
STA	A → M				8D	4	3	85	3	2							81	6	2	91	6	2	95	4	2	9D	5	3	99	5	3										· · · · · · · ·	STA
STX	X → M				8E	4	3	86	3	2																															· · · · · · · ·	STX
STY	Y → M				8C	4	3	84	3	2													94	4	2																	· · · · · · · ·	STY
TAX	A → X													AA	2	1																									N · · · · · Z ·	TAX
TAY	A → Y													A8	2	1																									N · · · · · Z ·	TAY
TSX	S → X													BA	2	1																									N · · · · · Z ·	TSX
TXA	X → A													8A	2	1																									N · · · · · Z ·	TXA
TXS	X → S													9A	2	1																									· · · · · · · ·	TXS
TYA	Y → A													98	2	1																									N · · · · · Z ·	TYA

(1) ADD 1 to "N" IF PAGE BOUNDARY IS CROSSED
(2) ADD 1 TO "N" IF BRANCH OCCURS TO SAME PAGE
ADD 2 TO "N" IF BRANCH OCCURS TO DIFFERENT PAGE
(3) CARRY NOT = BORROW
(4) IF IN DECIMAL MODE, Z FLAG IS INVALID
ACCUMULATOR MUST BE CHECKED FOR ZERO RESULT

X	INDEX X	+	ADD	M_7	MEMORY BIT 7
Y	INDEX Y	−	SUBTRACT	M_6	MEMORY BIT 6
A	ACCUMULATOR	∧	AND	n	NO. CYCLES
M	MEMORY PER EFFECTIVE ADDRESS	∨	OR	#	NO. BYTES
Ms	MEMORY PER STACK POINTER	∨	EXCLUSIVE OR		

Tabla 6-1.—Cuadro resumido de las instrucciones de los microprocesadores R 6500, ofrecido por el propio fabricante.

2.ª PARTE: JUEGO DE INSTRUCCIONES DE LA CPU MC-6800

MODOS DE DIRECCIONAMIENTO

El microprocesador 6800 de Motorola dispone de unas 100 instrucciones, que se convierten en unos 200 códigos diferentes, teniendo en cuenta los diversos modos de direccionado que admite cada instrucción. Este microprocesador dispone de 6 posibles modos de direccionado:

1.° *Inmediato.*—Cuando el operando o el dato que maneja la instrucción viene definido en el 2.° byte, detrás del que expresa el código OP.

2.° *Directo.*—Cuando el operando está almacenado en una posición de memoria de una determinada página, que define el byte siguiente al correspondiente del código OP. Mediante este modo de direccionado sólo se pueden controlar 256 posiciones de una página fija. Este direccionado también se llama "paginado".

3.° *Indexado.*—El 2.° byte, que sigue al código OP, es un offset, que ha de sumarse a los 8 bits de menos peso del registro Indice. El resultado de la suma, que representa la dirección efectiva donde está el operando, se almacena en un registro Temporal de 16 bits para que no altere el contenido del registro Indice. Se trata del modo de direccionado más potente que posee el 6800.

4.° *Extendido.*—El 2.° y 3.° byte, que sigue al código OP de la instrucción, determinan la dirección completa (página y posición) en donde se encuentra almacenado el operando.

5.° *Inherente o Implícito.*—El dato u operando va implícito con la instrucción a que acompaña.

6.° *Relativo.*—El 2.° byte es un offset, que se suma a los 8 bits de menos peso del PC más 2 para obtener la dirección absoluta. Se emplea en las instrucciones de bifurcación.

CUADRO RESUMIDO DEL REPERTORIO DE INSTRUCCIONES DE LA CPU 6800

Una vez analizadas con cierto detalle las instrucciones de las CPU de la familia R 6500, debido a la semejanza que existe con las de los demás miroprocesadores, las correspondientes al 6800 sólo se indican de forma resumida, expresando su función de forma gráfica. En cuanto a los símbolos que se emplearán, también son muy parecidos a los del 6500 y sólo debe tenerse en cuenta que el microprocesador MC-6800 dispone de

2 Acumuladores llamados A y B, que el registro de Estado se representará por CCR, el puntero del stack por SP (SP_H y SP_L) y su contenido por M_{SP}.

Se recomienda acudir a los manuales de información técnica de los propios fabricantes, que en este caso particular son muy completos, para obtener una información más detallada y extensa.

REPERTORIO DE INSTRUCCIONES DEL MC-6800

Operación	Nemónico	Expresión Gráfica	Código OP según direccionamiento				
			Inmediato	Directo	Indexado	Extendido	Inherente
Instrucciones con los acumuladores y la memoria							
Suma	ADDA ADDB	A + M → A B + M → B	8B CB	9B DB	AB EB	BB FB	
Suma con acarreo	ADCA ADCB	A + M + C → A B + M + C → B	89 C9	99 D9	A9 E9	B9 F9	
Suma de Acumuladores	ABA	A + B → A					1B
Operación AND	ANDA ANDB	A ∧ M → A B ∧ M → B	84 C4	94 D4	A4 E4	B4 F4	
Testado de bits	BITA BITB	A ∧ M B ∧ M	85 C5	95 D5	A5 E5	B5 F5	
Borrado o puesta a 0	CLR CLRA CLRB	OO → M OO → A OO → B			6F	7F	 4F 5F
Comparación	CMPA CMPB	A − M B − M	81 C1	91 D1	A1 E1	B1 F1	
Complemento a 1	COM COMA COMB	$\overline{M}$ → M $\overline{A}$ → A $\overline{B}$ → B			63	73	 43 53
Complemento a 2	NEG NEGA NEGB	OO − M → M OO − A → A OO − B → B			60	70	 40 50

REPERTORIO DE INSTRUCCIONES DEL MC-6800 (Continuación)

Operación	Nemónico	Expresión Gráfica	Código OP según direccionamiento				
			Inmediato	Directo	Indexado	Extendido	Inherente
Comparación de Acumuladores	CBA	A − B					11
Ajuste decimal, A	DDA	Convierte la suma binaria en formato BCD.					19
Decrementos	DEC DECA DECB	M − 1 → M A − 1 → A B − 1 → B			6A	7A	 4A 5A
Exclusiva OR	EORA EORB	A ∀ M → A A ∀ M → A	88 C8	98 D8	A8 E8	B8 F8	
Incrementos	INC INCA INCB	M + 1 → M A + 1 → A B + 1 → B			6C	7C	 4C 5C
Cargar Acumulador	LDAA LDAB	M → A M → B	86 C6	96 D6	A6 E6	B6 F6	
Operación OR	ORA A ORA B	A ∨ M → A B ∨ M → B	8A CA	9A DA	AA EA	BA FA	
Guardar datos	PSHA PSHB	A → M_{SP}, SP − 1 → SP B → M_{SP}, SP − 1 → SP				36 37	
Extraer datos	PULA PULB	SP + 1 → SP, M_{SP} → A SP + 1 → SP, M_{SP} → B				32 33	
Rotación izquierda	ROL ROLA ROLB	M A B			69	79	 49 59
Rotación derecha	ROR RORA RORB	M A B			66	76	 46 56

REPERTORIO DE INSTRUCCIONES DEL MC-6800 (Continuación)

Operación	Nemónico	Expresión Gráfica	Código OP según direccionamiento				
			Inmediato	Directo	Indexado	Extendido	Inherente
Desplazam. aritmético a la izq.	ASL ASLA ASLB	M A B			68	78	 48 58
Desplazam. aritmético a la derecha	ASR ASRA ASRB	M A B			67	77	 47 57
Desplazam. derecha lógico	LSR LSRA LSRB	M A B			64	74	 44 54
Almacenar Acumulador	STAA STAB	A → M B → M		97 D7	A7 E7	B7 F7	
Resta	SUBA SUBB	A − M → A A − M → B	80 C0	90 D0	A0 E0	B0 F0	
Resta de Acumuladores	SBA	A − B → A					10
Resta con carry	SBCA SBCB	A − M − C → A B − M − C → B	82 C2	92 D2	A2 E2	B2 F2	
Transferencia de acumuladores	TAB TBA	A → B B → A					16 17
Test, cero ó menos	TST TSTA TSTB	M-OO A-OO B-OO			6D	7D	 4D 5D
Compara el registro Indice	CPX	(XH/XL) − − (M/M + 1)	8C	9C	AC	BC	
Decrementa el registro Indice	DEX	X − 1 → X					09
Decrementa el stack	DES	SP − 1 → SP					34

REPERTORIO DE INSTRUCCIONES DEL MC-6800 (Continuación)

Operación	Nemónico	Expresión Gráfica	Código OP según direccionamiento				
			Inmediato	Directo	Indexado	Extendido	Inherente
Incrementa el registro Indice	INX	$X + 1 \rightarrow X$					08
Incrementa el stack	INS	$SP + 1 \rightarrow SP$					31
Carga, el registro Indice	LDX	$M \rightarrow X_H$; $(M + 1) \rightarrow X_L$	CE	DE	EE	FE	
Carga el stack	LDS	$M \rightarrow SP_H$; $(M + 1) \rightarrow SP_L$	8E	9E	AE	BE	
Almacena el registro Indice	STX	$X_H \rightarrow M$; $X_L \rightarrow M + 1$		DF	EF	FF	
Almacena el stack	STS	$SP_H \rightarrow M$; $SP_L \rightarrow M + 1$		9F	AF	BF	
Reg. Indice-Stack	TXS	$X - 1 \rightarrow SP$					35
Stack-Registro Indice	TSX	$SP + 1 \rightarrow X$					30
Instrucciones de Branch							
Branch siempre	BRA	Incondicional	20				
Branch, si carry es cero	BCC	Si C = 0	24				
Branch, si carry es uno	BCS	Si C = 1	25				
Branch, si Z = 1 (Resultado = 0) RES = 0	BEQ	Si Z = 1	27				

REPERTORIO DE INSTRUCCIONES DEL MC-6800 (Continuación)

Operación	Nemónico	Expresión Gráfica	Código OP según direccionamiento				
			Inmediato	Directo	Indexado	Extendido	Inherente
Branch, si RES $\geqslant 0$	BGE	Si N $\forall$ V = 0	26				
Branch, si RES > 0	BGT	Si Z $\wedge$ (N $\forall$ V) = 0	2E				
Branch, si RES es mayor	BHI	Si C $\wedge$ Z = 0	22				
Branch, si RES $\leqslant 0$	BLE	Si Z V (N $\forall$ V) = 1	2F				
Branch, si es menor o igual	BLS	Si C V Z = 1	23				
Branch, RES es menor	BLT	Si N $\forall$ V = 1	2D				
Branch, si RES es negativo	BMI	Si N = 1	2B				
Branch, si RES $\neq 0$	BNE	Si Z = 0	26				
Branch, si el overflow es cero	BVC	Si V = 0	28				
Branch, si el overflow es uno	BVS	Si V = 1	29				
Branch, si es positivo	BPL	Si N = 0	2A				

REPERTORIO DE INSTRUCCIONES DEL MC-6800 (Continuación)

Operación	Nemónico	Expresión Gráfica	Código OP según direccionamiento				
			Inmediato	Directo	Indexado	Extendido	Inherente
Branch a subrutina	BSR	Operaciones Especiales de rotura de secuencia.	8D				
Salto	JMP			6E	7E		
Salto a subrutina	JSR			AD	BD		
No operar	NOP	Adelanta solamente el programa contador					01
Retorno de interrupción	RTI	Operaciones Especiales					3B
Retorno para subrutina	RTS						39
Interrupción por software	SWI						3F
Esperar interrupción NMI ó IRQ deteniendo la CPU	WAI						3F
Borrar el Carry	CLC	O → C					0C
Borrar la máscara de interrupción	CLI	O → I					0E
Borrar el Overflow	CLV	O → V					0A
Poner a 1 (SET) el Carry	SEC	1 → C					0D

REPERTORIO DE INSTRUCCIONES DEL MC-6800 (Continuación)

Operación	Nemónico	Expresión Gráfica	Código OP Según direccionamiento				
			Inmediato	Directo	Indexado	Extendido	Inherente
Poner a 1 el flag Interrupción	SEI	1 → I					0F
Poner a 1 el flag Overflow	SEV	1 → V					0B
Transferencia del Acumulador al registro de Estado	TAP	A → CCR					06
Transferencia del registro de Estado al Acumulador	TPA	CCR → A					07

3.ª PARTE: JUEGO DE INSTRUCCIONES DE LA CPU 8080 DE INTEL (8085).

MODOS DE DIRECCIONAMIENTO

El microprocesador 8080 puede ser direccionado de 4 modos diferentes, que se explican a continuación:

1.° *Inmediato.*—El byte, que sigue al código OP de la instrucción, expresa el operando que maneja la instrucción.

2.° *Directo.*—Los 2 bytes, que siguen al código OP, proporcionan la dirección de la memoria donde se encuentra almacenado el operando.

3.° *Por registro.*—La instrucción determina cuál de los registros (B, C, D, E, H y L), bien individualmente o por parejas, contienen el dato u operando.

4.° *Por par de registros indirecto.*—La instrucción determina la pareja de registros: B—C, D—E o H—L, que contienen la dirección de la memoria en donde se encuentra el operando. En general, se suelen usar los registros H—L (que contienen los 8 bits de más peso y de menos peso respectivamente) para direccionar la memoria.

CUADRO RESUMIDO DEL JUEGO DE INSTRUCCIONES DE LA CPU 8080

El 8080 dispone de 78 instrucciones, que se exponen a continuación de forma resumida. Para comprender el significado de cada instrucción y la formación del código OP de cada una, se explican las abreviaturas empledas en la tabla.

r, r_1 y r_2: Se emplean para representar uno de los 8 registros, que son: A, B, C, D, E, H, L y M (dirección de memoria que determina el par de registros H y L).

Para la determinación de cada uno de los 8 registros comentados se emplea el siguiente código:

A: 111, B: 000, C: 001, D: 010, E: 011, H: 100, L: 101, y M: 110.

Cuando el registro actúa como destino de una información, se simbolizan los 3 bits, que lo codifican por DDD y, cuando actúa como fuente, por SSS.

rp: Representan a un par de registros conjuntamente. De forma que B representa a la pareja B—C, D a la D—E, H a la H—L y SP al stack.

Cada uno de estos 4 pares de registros se codifican mediante dos bits, llamados RP y que son los siguientes:

B: 00, D: 01, H: 10, y SP: 11.

rh: primer registro de un par.

rl: segundo registro, o registro de menos peso de un par.

n: representa un número del 0 al 7, de las 8 posiciones a las que puede saltar el PC, cuando se produce una instrucción RST. Se representan por NNN y sus valores llegan desde 000 hasta 111.

CCC: codifica la condición o estado de los flags, según el código siguiente:

CCC	
000	Z = 0 (Flag cero del Acumulador)
001	Z = 1
010	CY = 0 (Carry)
011	CY = 1
100	P = 0 (Flag de paridad)
101	P = 1
110	S = 0 (Flag de signo)
111	S = 1

Finalmente, como es fácil deducir, la abreviatura "mem" representa una dirección de una posición de memoria; "dato" representa un dato y "canal", la dirección de un canal de I/O.

INSTRUCCIONES DEL 8080

Nemónico	Expresión Gráfica	Código instrucción (1.º byte) OP							
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
MOV r_1, r_2	$r_1 \leftarrow r_2$ La información contenida en r_2 pasa al r_1	0	1	D	D	D	S	S	S
MOV r_1, dato	$r_1 \leftarrow$ dato El dato pasa a r_1	0	0	D	D	D	1	1	0
LXI rp, dato	rp ← dato El dato se coloca en el par de registros rp	0	0	R	P	0	0	0	1

INSTRUCCIONES DEL 8080 (Continuación)

Nemónico	Expresión Gráfica	Código instrucción (1.º byte) OP							
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
LDA mem.	A ← men El contenido de la posición direccionada de memoria se transfiere al acumulador	0	0	1	1	1	0	1	0
STA mem.	mem ← A El contenido del acumulador se guarda en una posición de memoria direccionada	0	0	1	1	0	0	1	0
LHLD mem.	HL ← mem El contenido que hay en una posición de memoria se coloca en el registro L. El contenido de esa dirección + 1, en el registro H	0	0	1	0	1	0	1	0
SHLD mem.	mem ← HL El contenido de los registro HL pasa a una posición de la memoria y a la siguiente	0	0	1	0	0	0	1	0
LDAX rP	A ← rP El contenido de la posición de memoria direccionada por el par de registros rP pasa al acumulador	0	0	R	P	1	0	1	0
STAX rP	rP ← A El contenido del acumulador se guarda en la posición de memoria direccionada por un par de registros	0	0	R	P	0	0	1	0
XCHG	H ↔ D L ↔ E El contenido de los registros H y L se intercambia por el contenido de los registros D y E respectivamente	1	1	1	0	1	0	1	1
ADD, r	A ← A + r Añadir el contenido del registro r al acumulador	1	0	0	0	0	S	S	S

INSTRUCCIONES DEL 8080 (Continuación)

Nemónico	Expresión Gráfica	Código instrucción (1.º byte) OP							
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
ADI, dato	A ← A + dato El dato ó 2.º byte se añade al acumulador	1	1	0	0	0	1	1	0
ADC, r	A ← A + r + CY Añadir el contenido del registro r y el arrastre CY al acumulador	1	0	0	0	1	S	S	S
ACI dato	A ← A + Dato + CY El dato y el arrastre se añaden al acumulador	1	1	0	0	1	1	1	0
SUB r	A ← A − r Sustraer el contenido del registro r, del acumulador	1	0	0	1	0	S	S	S
SUI dato	A ← A − dato El dato se resta del acumulador	1	1	0	1	0	1	1	0
SBBR	A ← A − r − CY Sustraer del acumulador el contenido del registro r y el arrastre	1	0	0	1	1	S	S	S
SBI dato	A ← A − dato − CY El dato y el arrastre se restan del acumulador	1	1	0	1	1	1	1	0
INR r	r ← r + 1 Incrementar el contenido del registro r	0	0	D	D	D	1	0	0
DCR r	r ← r − 1 Decrementar el contenido del registro r	0	0	D	D	D	1	0	1
INX rP	El contenido del par de registros rp se incrementa en una unidad rp ← rp + 1	0	0	R	P	O	O	1	1
DCX rP	rp ← rp − 1 El contenido del par de registros rp se decrementa en una unidad	0	0	R	P	1	0	1	1

INSTRUCCIONES DEL 8080 (Continuación)

Nemónico	Expresión Gráfica	Código instrucción (1.º byte) OP							
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
DAD rp	HL ← rp + HL El contenido del par de registros rp se añade al contenido del par de registros H y L. El resultado permanece en H y L	0	0	R	P	1	0	0	1
DAA	Ajuste decimal en el acumulador formando 2 dígitos de 4 bits en BCD	0	0	1	0	0	1	1	1
ANA r	A ← A ∧ r Ejecutar la intersección lógica entre el registro r y el acumulador. Operación AND	1	0	1	0	0	S	S	S
ANI dato	A ← A ∧ dato Intersección lógica entre el dato y el acumulador	1	1	1	0	0	1	1	0
XRA r	A ← A ∀ r Efectuar la operación lógica 0 - Exclusiva entre el registro r y el acumulador	1	0	1	0	1	S	S	S
XRI dato	A ← A ∀ dato Operación 0 - Exclusiva entre el dato y el acumulador	1	1	1	0	1	1	1	0
ORA r	A ← A ∨ r Reunión lógica entre el contenido del registro r y el acumulador. El Resultado queda en el acumulador. Operación OR	1	0	1	1	0	S	S	S
ORI dato	A ← A ∨ dato Reunión lógica entre el dato y el acumulador	1	0	1	1	1	S	S	S
CMP r	A − r (resta) Compara el contenido del registro r con el acumulador. El acumulador no cambia. El flag Z = 1, si A = r. El flag CY = 1, si A < r	1	0	1	1	1	S	S	S

INSTRUCCIONES DEL 8080 (Continuación)

Nemónico	Expresión Gráfica	Código instrucción (1.º byte) OP							
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
CPI dato	A – dato Comparación del dato con el acumulador. El acumulador no cambia	1	1	1	1	1	1	1	0
RLC	Rotación del acumulador a la izquierda $(A_{n+1}) \leftarrow (A_n)$; $(A_0) \leftarrow (A_7)$; $(CY) \leftarrow (A_7)$	0	0	0	0	0	1	1	1
RRC	Rotación del acumulador a la derecha $(A_n) \leftarrow (A_{n+1})$; $(A_7) \leftarrow (A_0)$; $(CY) \leftarrow (A_0)$	0	0	0	0	1	1	1	1
RAL	Rotación del acumulador a la izquierda a través del carry $(A_{n+1}) \leftarrow (A_n)$; $(CY) \leftarrow (A_7)$; $(A_0) \leftarrow (CY)$	0	0	0	1	0	1	1	1
RAR	Rotación del acumulador a la derecha a través del carry $(A_n) \leftarrow (A_{n+1})$; $(CY) \leftarrow (A_0)$; $(A_7) \leftarrow (CY)$	0	0	0	1	1	1	1	1
CMA	$A \leftarrow \overline{A}$ Complementar el acumulador	0	0	1	0	1	1	1	1
CMC	$CY \leftarrow \overline{CY}$ Arrastre complementado	0	0	1	1	1	1	1	1
STC	$CY \leftarrow 1$ Arrastre igual a 1	0	0	1	1	0	1	1	1
JMP mem	PC ← mem El contador de programa queda cargado con la dirección de memoria dada por el 2.º y 3.º bytes	1	1	0	0	0	0	1	1
JXX mem	PC ← mem; Si CCC = 1 Depende de los flags. Salto condicional	1	1	C	C	C	0	1	0

INSTRUCCIONES DEL 8080 (Continuación)

Nemónico	Expresión Gráfica	Código instrucción (1.º byte) OP							
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
CALL mem	SP − 1 ← PCH SP − 2 ← PCL SP ← SP − 2 PC ← mem (3.º byte, 2.º byte) Se salta incondicionalmente a la subrutina que empieza en mem, dada por el 2.º y 3.º bytes	1	1	0	0	1	1	0	1
CXX mem Condicional	Igual a CALL si ccc se cumple. Depende de los flags	1	1	C	C	C	1	0	0
RET	PCL ← SP PCH ← SP + 1 SP ← SP + 2 Vuelve incondiconalmente al programa del que se procedía	1	1	0	0	1	0	0	1
RXX	Igual a RET si CCC = 1. Depende de los flags	1	1	C	C	C	0	0	0
RST n	Salto incondicional a una subrutina cuya dirección es n. SP − 1 ← PCH SP − 2 ← PCL SP ← SP − 2 PC ← 8 *NNN	1	1	N	N	N	1	1	1
PCHL	PC ← HL El contenido de HL, reemplaza al contenido del contador de programa	1	1	1	0	1	0	0	1
PUSH rp	SP − 1 ← rh SP − 2 ← rl SP ← SP − 2 El contenido del par de registros rp, se coloca en el stack	1	1	R	P	0	1	0	1
POP rp	rl ← SP rh ← SP + 1 SP ← SP + 2 El contenido del stack se traspasa al par de registros rp	1	1	R	P	0	0	0	1

INSTRUCCIONES DEL 8080 (Continuación)

Nemónico	Expresión Gráfica	Código instrucción (1.º byte) OP							
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
PUSH PSW	El contenido del registro P, pasa al stack	1	1	1	1	0	1	0	1
POP PSW	El contenido del stack pasa al registro P	1	1	1	1	0	0	0	1
XTHL	L ↔ SP H ↔ SP + 1 El contenido de los registros L y H se intercambia con las 2 primeras posiciones del Stack	1	1	1	0	0	0	1	1
SPHL	SP ← HL El contenido de H y L reemplaza el contenido del puntero del stack	1	1	1	1	1	0	0	1
IN canal	A ← (datos canal) La entrada "canal", suministra sus 8 bits de información, que quedan depositados en el acumulador	1	1	0	1	1	0	1	
OUT canal	El contenido del acumulador se envía al canal de salida: (datos canal) → A	1	1	0	1	0	0	1	1
EI	Interrupciones abiertas. El flag de interrupción pasa al estado de habilitación, permitiendo que la CPU reconozca y responda a una interrupción	1	1	1	1	1	0	1	1
DI	Interrupciones bloqueadas. El flag de interrupción pasa al estado de prohibición, con lo que la CPU ignora las interrupciones posteriores	1	1	1	1	0	0	1	1
HLT	Alto, stop. La CPU entra en un estado de parada, del que sólo puede salir a través de una interrupción	0	1	1	1	0	1	1	0

INSTRUCCIONES DEL 8080 (Continuación)

Nemónico	Expresión Gráfica	Código instrucción (1.º byte) OP							
		D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
NOP	No operar. La instrucción no modifica nada en absoluto y se pasa a la siguiente	0	0	0	0	0	0	0	0

Nota: Existen también las siguientes instrucciones: Mov r, mem; Mov mem, r; ADD mem; ADC mem; SUB mem; SBB mem; INR mem; DCR mem; ANA mem; ORA mem; CMP mem; que son similares a otras ya comentadas y que se referían a registros o datos. En este tipo de instrucciones participa, como uno de los operandos, el contenido de la posición de memoria, definida por el par de registros H y L.

JUEGO DE INSTRUCCIONES DEL 8085

A continuación, y con carácter resumido, se ofrece el juego de instrucciones del 8085, en el que se aprecia que coincide completamente con el del 8080, a excepción de 2 instrucciones específicas para el 8085: RIM y SIM, que ya fueron comentadas en el capítulo precedente.

Todos los nemónicos tienen copyright de INTEL Corporation, 1977.

La primera columna de la siguiente tabla expresa el código nemónico y la segunda la descripción de la función que realiza cada instrucción. A continuación se ofrecen los 8 bits que forman el código máquina y finalmente los ciclos.

Nota 1.—DDD o SSS: B: 000, C: 001, D: 010, E: 011, H: 100, L: 101, Memoria: 111.

Nota 2.—Existen dos posibles ciclos de tiempo /6/12), los ciclos que refleja cada instrucción dependen de los flags de condición.

JUEGO DE INSTRUCCIONES DEL 8085

Mnemonic	Description	Instruction Code[1] D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Clock[2] Cycles
MOVE, LOAD, AND STORE										
MOV r1,r2	Move register to register	0	1	D	D	D	S	S	S	4
MOV M,r	Move register to memory	0	1	1	1	0	S	S	S	7
MOV r,M	Move memory to register	0	1	D	D	D	1	1	0	7
MVI r	Move immediate register	0	0	D	D	D	1	1	0	7
MVI M	Move immediate memory	0	0	1	1	0	1	1	0	10
LXI B	Load immediate register Pair B & C	0	0	0	0	0	0	0	1	10
LXI D	Load immediate register Pair D & E	0	0	0	1	0	0	0	1	10
LXI H	Load immediate register Pair H & L	0	0	1	0	0	0	0	1	10
LXI SP	Load immediate stack pointer	0	0	1	1	0	0	0	1	10
STAX B	Store A indirect	0	0	0	0	0	0	1	0	7
STAX D	Store A indirect	0	0	0	1	0	0	1	0	7
LDAX B	Load A indirect	0	0	0	0	1	0	1	0	7
LDAX D	Load A indirect	0	0	0	1	1	0	1	0	7
STA	Store A direct	0	0	1	1	0	0	1	0	13
LDA	Load A direct	0	0	1	1	1	0	1	0	13
SHLD	Store H & L direct	0	0	1	0	0	0	1	0	16
LHLD	Load H & L direct	0	0	1	0	1	0	1	0	16
XCHG	Exchange D & E, H & L Registers	1	1	1	0	1	0	1	1	4
STACK OPS										
PUSH B	Push register Pair B & C on stack	1	1	0	0	0	1	0	1	12
PUSH D	Push register Pair D & E on stack	1	1	0	1	0	1	0	1	12
PUSH H	Push register Pair H & L on stack	1	1	1	0	0	1	0	1	12
PUSH PSW	Push A and Flags on stack	1	1	1	1	0	1	0	1	12
POP B	Pop register Pair B & C off stack	1	1	0	0	0	0	0	1	10
POP D	Pop register Pair D & E off stack	1	1	0	1	0	0	0	1	10
POP H	Pop register Pair H & L off stack	1	1	1	0	0	0	0	1	10
POP PSW	Pop A and Flags off stack	1	1	1	1	0	0	0	1	10
XTHL	Exchange top of stack, H & L	1	1	1	0	0	0	1	1	16
SPHL	H & L to stack pointer	1	1	1	1	1	0	0	1	6
JUMP										
JMP	Jump unconditional	1	1	0	0	0	0	1	1	10
JC	Jump on carry	1	1	0	1	1	0	1	0	7/10
JNC	Jump on no carry	1	1	0	1	0	0	1	0	7/10
JZ	Jump on zero	1	1	0	0	1	0	1	0	7/10
JNZ	Jump on no zero	1	1	0	0	0	0	1	0	7/10
JP	Jump on positive	1	1	1	1	0	0	1	0	7/10
JM	Jump on minus	1	1	1	1	1	0	1	0	7/10
JPE	Jump on parity even	1	1	1	0	1	0	1	0	7/10
JPO	Jump on parity odd	1	1	1	0	0	0	1	0	7/10
PCHL	H & L to program counter	1	1	1	0	1	0	0	1	6
CALL										
CALL	Call unconditional	1	1	0	0	1	1	0	1	18
CC	Call on carry	1	1	0	1	1	1	0	0	9/18
CNC	Call on no carry	1	1	0	1	0	1	0	0	9/18
CZ	Call on zero	1	1	0	0	1	1	0	0	9/18
CNZ	Call on no zero	1	1	0	0	0	1	0	0	9/18
CP	Call on positive	1	1	1	1	0	1	0	0	9/18
CM	Call on minus	1	1	1	1	1	1	0	0	9/18
CPE	Call on parity even	1	1	1	0	1	1	0	0	9/18
CPO	Call on parity odd	1	1	1	0	0	1	0	0	9/18
RETURN										
RET	Return	1	1	0	0	1	0	0	1	10
RC	Return on carry	1	1	0	1	1	0	0	0	6/12
RNC	Return on no carry	1	1	0	1	0	0	0	0	6/12
RZ	Return on zero	1	1	0	0	1	0	0	0	6/12
RNZ	Return on no zero	1	1	0	0	0	0	0	0	6/12
RP	Return on positive	1	1	1	1	0	0	0	0	6/12
RM	Return on minus	1	1	1	1	1	0	0	0	6/12
RPE	Return on parity even	1	1	1	0	1	0	0	0	6/12
RPO	Return on parity odd	1	1	1	0	0	0	0	0	6/12
RESTART										
RST	Restart	1	1	A	A	A	1	1	1	12
INPUT/OUTPUT										
IN	Input	1	1	0	1	1	0	1	1	10
OUT	Output	1	1	0	1	0	0	1	1	10
INCREMENT AND DECREMENT										
INR r	Increment register	0	0	D	D	D	1	0	0	4
DCR r	Decrement register	0	0	D	D	D	1	0	1	4
INR M	Increment memory	0	0	1	1	0	1	0	0	10
DCR M	Decrement memory	0	0	1	1	0	1	0	1	10
INX B	Increment B & C registers	0	0	0	0	0	0	1	1	6
INX D	Increment D & E registers	0	0	0	1	0	0	1	1	6
INX H	Increment H & L registers	0	0	1	0	0	0	1	1	6
INX SP	Increment stack pointer	0	0	1	1	0	0	1	1	6
DCX B	Decrement B & C	0	0	0	0	1	0	1	1	6
DCX D	Decrement D & E	0	0	0	1	1	0	1	1	6
DCX H	Decrement H & L	0	0	1	0	1	0	1	1	6
DCX SP	Decrement stack pointer	0	0	1	1	1	0	1	1	6
ADD										
ADD r	Add register to A	1	0	0	0	0	S	S	S	4
ADC r	Add register to A with carry	1	0	0	0	1	S	S	S	4
ADD M	Add memory to A	1	0	0	0	0	1	1	0	7
ADC M	Add memory to A with carry	1	0	0	0	1	1	1	0	7
ADI	Add immediate to A	1	1	0	0	0	1	1	0	7
ACI	Add immediate to A with carry	1	1	0	0	1	1	1	0	7
DAD B	Add B & C to H & L	0	0	0	0	1	0	0	1	10
DAD D	Add D & E to H & L	0	0	0	1	1	0	0	1	10
DAD H	Add H & L to H & L	0	0	1	0	1	0	0	1	10
DAD SP	Add stack pointer to H & L	0	0	1	1	1	0	0	1	10
SUBTRACT										
SUB r	Subtract register from A	1	0	0	1	0	S	S	S	4
SBB r	Subtract register from A with borrow	1	0	0	1	1	S	S	S	4
SUB M	Subtract memory from A	1	0	0	1	0	1	1	0	7
SBB M	Subtract memory from A with borrow	1	0	0	1	1	1	1	0	7
SUI	Subtract immediate from A	1	1	0	1	0	1	1	0	7
SBI	Subtract immediate from A with borrow	1	1	0	1	1	1	1	0	7
LOGICAL										
ANA r	And register with A	1	0	1	0	0	S	S	S	4

JUEGO DE INSTRUCCIONES DEL 8085 (Continuación)

Mnemonic	Description	Instruction Code[1] D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Clock[2] Cycles
XRA r	Exclusive Or register with A	1	0	1	0	1	S	S	S	4
ORA r	Or register with A	1	0	1	1	0	S	S	S	4
CMP r	Compare register with A	1	0	1	1	1	S	S	S	4
ANA M	And memory with A	1	0	1	0	0	1	1	0	7
XRA M	Exclusive Or memory with A	1	0	1	0	1	1	1	0	7
ORA M	Or memory with A	1	0	1	1	0	1	1	0	7
CMP M	Compare memory with A	1	0	1	1	1	1	1	0	7
ANI	And immediate with A	1	1	1	0	0	1	1	0	7
XRI	Exclusive Or immediate with A	1	1	1	0	1	1	1	0	7
ORI	Or immediate with A	1	1	1	1	0	1	1	0	7
CPI	Compare immediate with A	1	1	1	1	1	1	1	0	7
ROTATE										
RLC	Rotate A left	0	0	0	0	0	1	1	1	4
RRC	Rotate A right	0	0	0	0	1	1	1	1	4
RAL	Rotate A left through carry	0	0	0	1	0	1	1	1	4
RAR	Rotate A right through carry	0	0	0	1	1	1	1	1	4
SPECIALS										
CMA	Complement A	0	0	1	0	1	1	1	1	4
STC	Set carry	0	0	1	1	0	1	1	1	4
CMC	Complement carry	0	0	1	1	1	1	1	1	4
DAA	Decimal adjust A	0	0	1	0	0	1	1	1	4
CONTROL										
EI	Enable Interrupts	1	1	1	1	1	0	1	1	4
DI	Disable Interrupt	1	1	1	1	0	0	1	1	4
NOP	No-operation	0	0	0	0	0	0	0	0	4
HLT	Halt	0	1	1	1	0	1	1	0	5
NEW 8085 INSTRUCTIONS										
RIM	Read Interrupt Mask	0	0	1	0	0	0	0	0	4
SIM	Set Interrupt Mask	0	0	1	1	0	0	0	0	4

Finalmente, se presenta un cuadro con las instrucciones del 8085, clasificadas en cuatro grupos según sus funciones: 1.°) Transferencia de datos, 2.°) Operaciones lógicas y aritméticas, 3.°) Branch (bifurcación o salto) y derivadas y 4.°) De entradas/salidas y control.

GRUPO DE TRANSFERENCIA DE DATOS

Move

MOV	A,A	7F
	A,B	78
	A,C	79
	A,D	7A
	A,E	7B
	A,H	7C
	A,L	7D
	A,M	7E
MOV	B,A	47
	B,B	40
	B,C	41
	B,D	42
	B,E	43
	B,H	44
	B,L	45
	B,M	46
MOV	C,A	4F
	C,B	48
	C,C	49
	C,D	4A
	C,E	4B
	C,H	4C
	C,L	4D
	C,M	4E
MOV	D,A	57
	D,B	50
	D,C	51
	D,D	52
	D,E	53
	D,H	54
	D,L	55
	D,M	56

Move (cont)

MOV	E,A	5F
	E,B	58
	E,C	59
	E,D	5A
	E,E	5B
	E,H	5C
	E,L	5D
	E,M	5E
MOV	H,A	67
	H,B	60
	H,C	61
	H,D	62
	H,E	63
	H,H	64
	H,L	65
	H,M	66
MOV	L,A	6F
	L,B	68
	L,C	69
	L,D	6A
	L,E	6B
	L,H	6C
	L,L	6D
	L,M	6E
MOV	M,A	77
	M,B	70
	M,C	71
	M,D	72
	M,E	73
	M,H	74
	M,L	75

XCHG EB

Move Immediate

MVI	A, byte	3E
	B, byte	06
	C, byte	0E
	D, byte	16
	E, byte	1E
	H, byte	26
	L, byte	2E
	M, byte	36

Load Immediate

LXI	B, dble	01
	D, dble	11
	H, dble	21
	SP, dble	31

Load/Store

LDAX B	0A
LDAX D	1A
LHLD adr	2A
LDA adr	3A
STAX B	02
STAX D	12
SHLD adr	22
STA adr	32

byte = constant, or logical/arithmetic expression that evaluates to an 8-bit data quantity. (Second byte of 2-byte instructions)

dble = constant, or logical/arithmetic expression that evaluates to a 16-bit data quantity. (Second and Third bytes of 3-byte instructions)

adr = 16-bit address (Second and Third bytes of 3-byte instructions)

* = all flags (C, Z, S, P, AC) affected.

** = all flags except CARRY affected, (exception: INX and DCX affect no flags)

† = only CARRY affected.

GRUPO DE LOGICAS Y ARITMETICAS

Add*

ADD	A	87
	B	80
	C	81
	D	82
	E	83
	H	84
	L	85
	M	86
ADC	A	8F
	B	88
	C	89
	D	8A
	E	8B
	H	8C
	L	8D
	M	8E

Subtract*

SUB	A	97
	B	90
	C	91
	D	92
	E	93
	H	94
	L	95
	M	96
SBB	A	9F
	B	98
	C	99
	D	9A
	E	9B
	H	9C
	L	9D
	M	9E

Double Add †

DAD	B	09
	D	19
	H	29
	SP	39

Increment**

INR	A	3C
	B	04
	C	0C
	D	14
	E	1C
	H	24
	L	2C
	M	34
INX	B	03
	D	13
	H	23
	SP	33

Decrement**

DCR	A	3D
	B	05
	C	0D
	D	15
	E	1D
	H	25
	L	2D
	M	35
DCX	B	0B
	D	1B
	H	2B
	SP	3B

Specials

DAA*	27
CMA	2F
STC†	37
CMC†	3F

Rotate †

RLC	07
RRC	0F
RAL	17
RAR	1F

Logical*

ANA	A	A7
	B	A0
	C	A1
	D	A2
	E	A3
	H	A4
	L	A5
	M	A6
XRA	A	AF
	B	A8
	C	A9
	D	AA
	E	AB
	H	AC
	L	AD
	M	AE
ORA	A	B7
	B	B0
	C	B1
	D	B2
	E	B3
	H	B4
	L	B5
	M	B6
CMP	A	BF
	B	B8
	C	B9
	D	BA
	E	BB
	H	BC
	L	BD
	M	BE

Arith & Logical Immediate

ADI byte	C6
ACI byte	CE
SUI byte	D6
SBI byte	DE
ANI byte	E6
XRI byte	EE
ORI byte	F6
CPI byte	FE

B RANCH

Jump

JMP adr	C3
JNZ adr	C2
JZ adr	CA
JNC adr	D2
JC adr	DA
JPO adr	E2
JPE adr	EA
JP adr	F2
JM adr	FA
PCHL	E9

Call

CALL adr	CD
CNZ adr	C4
CZ adr	CC
CNC adr	D4
CC adr	DC
CPO adr	E4
CPE adr	EC
CP adr	F4
CM adr	FC

Return

RET	C9
RNZ	C0
RZ	C8
RNC	D0
RC	D8
RPO	E0
RPE	E8
RP	F0
RM	F8

Restart

RST	0	C7
	1	CF
	2	D7
	3	DF
	4	E7
	5	EF
	6	F7
	7	FF

I/O Y CONTROL

Stack Ops

PUSH	B	C5
	D	D5
	H	E5
	PSW	F5
POP	B	C1
	D	D1
	H	E1
	PSW*	F1
XTHL		E3
SPHL		F9

Input/Output

OUT byte	D3
IN byte	DB

Control

DI	F3
EI	FB
NOP	00
HLT	76

New Instructions (8085 Only)

RIM	20
SIM	30

CICLOS DE EJECUCION Y CODIGO DE LAS INSTRUCCIONES

Se detallan los ciclos de ejecución de las instrucciones del 8080, así como los códigos de las mismas.

El tiempo de ejecución de una instrucción es igual al número de período de la misma, multiplicado por el tiempo de duración de un período. La duración de éste puede variar entre 480 nanosegundos y 2 microsegundos.

Cuando en una instrucción se especifican dos números de períodos (por ejemplo 5/11), el número menor es el número de períodos de la misma si no se cumple la condición especificada, mientras que el número mayor es el que se realiza cuando la condición es verdadera.

DECIMAL	OCTAL	HEXADECIMAL	INSTRUCCION	NUMERO PERIODOS
0	000	00	NOP	4
1	001	01	LXI B	10
2	002	02	XTAX B	7
3	003	03	INX B	5
4	004	04	INR B	5
5	005	05	DCR B	5
6	006	06	MVI B	7
7	007	07	RLC	4
8	010	08	–	–
9	011	09	DAD B	10
10	012	0A	LDAX B	7
11	013	0B	DCX B	5
12	014	0C	INR C	5
13	015	0D	DCR C	5
14	016	0E	MVI C	7
15	017	0F	RRC	4
16	020	10	–	–
17	021	11	LXI D	10
18	022	12	STAX D	7
19	023	13	INX D	5
20	024	14	INR D	5
21	025	15	DCR D	5
22	026	16	MVI D	7
23	027	17	RAL	4
24	030	18	–	–
25	031	19	DAD D	10
26	032	1A	LDAX D	7
27	033	1B	DCX D	5
28	034	1C	INR E	5
29	035	1D	DCR E	5
30	036	1E	MVI E	7
31	037	1F	RAR	4

DECIMAL	OCTAL	HEXADECIMAL	INSTRUCCION	NUMERO PERIODOS
32	040	20	–	–
33	041	21	LXI H	10
34	042	22	SHLD	16
35	043	23	INX H	5
36	044	24	INR H	5
37	045	25	DCR H	5
38	046	26	MVI H	7
39	047	27	DAA	4
40	050	28	–	–
41	051	29	DAD H	10
42	052	2A	LHLD	16
43	053	2B	DCX H	5
44	054	2C	INR L	5
45	055	2D	DCR L	5
46	056	2E	MVI L	7
47	057	2F	CMA	4
48	060	30	–	–
49	061	31	LXI SP	10
50	062	32	STA	13
51	063	33	INX SP	5
52	064	34	INR M	10
53	065	35	DCR M	10
54	066	36	MVI M	10
55	067	37	STC	4
56	070	38	–	–
57	071	39	DAD SP	10
58	072	3A	LDA	13
59	073	3B	DCX SP	5
60	074	3C	INR A	5
61	075	3D	DCR A	5
62	076	3E	MVI A	5
63	077	3F	CMC	4
64	100	40	MOV B,B	5
65	101	41	MOV B,C	5
66	102	42	MOV B,D	5
67	103	43	MOV B,E	5
68	104	44	MOV B,H	5
69	105	45	MOV B,L	5
70	106	46	MOV B,M	7
71	107	47	MOV B,A	5
72	110	48	MOV C,B	5
73	111	49	MOV C,C	5
74	112	4A	MOV C,D	5
75	113	4B	MOV C,E	5
76	114	4C	MOV C,H	5
77	115	4D	MOV C,L	5
78	116	4E	MOV C,M	7
79	117	4F	MOV C,A	5
80	120	50	MOV D,B	5
81	121	51	MOV D,C	5

DECIMAL	OCTAL	HEXADECIMAL	INSTRUCCION	NUMERO PERIODOS
82	122	52	MOV D,D	5
83	123	53	MOV D,E	5
84	124	54	MOV D,H	5
85	125	55	MOV D,L	5
86	126	56	MOV D,M	7
87	127	57	MOV D,A	5
88	130	58	MOV E,B	5
89	131	59	MOV E,C	5
90	132	5A	MOV E,D	5
91	133	5B	MOV E,E	5
92	134	5C	MOV E,H	5
93	135	5D	MOV E,L	5
94	136	5E	MOV E,M	7
95	137	5F	MOV E,A	5
96	140	60	MOV H,B	5
97	141	61	MOV H,C	5
98	142	62	MOV H,D	5
99	143	63	MOV H,E	5
100	144	64	MOV H,H	5
101	145	65	MOV H,L	5
102	146	66	MOV H,M	7
103	147	67	MOV H,A	5
104	150	68	MOV L,B	5
105	151	69	MOV L,C	5
106	152	6A	MOV L,D	5
107	153	6B	MOV L,E	5
108	154	6C	MOV L,H	5
109	155	6D	MOV L,L	5
110	156	6E	MOV L,M	7
111	157	6F	MOV L,A	5
112	160	70	MOV M,B	7
113	161	71	MOV M,C	7
114	162	72	MOV M,D	7
115	163	73	MOV M,E	7
116	164	74	MOV M,H	7
117	165	75	MOV M,L	7
118	166	76	HLT	7
119	167	77	MOV M,A	7
120	170	78	MOV A,B	5
121	171	79	MOV A,C	5
122	172	7A	MOV A,D	5
123	173	7B	MOV A,E	5
124	174	7C	MOV A,H	5
125	175	7D	MOV A,L	5
126	176	7E	MOV A,M	7
127	177	7F	MOV A,A	5
128	200	80	ADD B	4
129	201	81	ADD C	4
130	202	82	ADD D	4
131	203	83	ADD E	4

DECIMAL	OCTAL	HEXADECIMAL	INSTRUCCION	NUMERO PERIODOS
132	204	84	ADD H	4
133	205	85	ADD L	4
134	206	86	ADD M	7
135	207	87	ADD A	4
136	210	88	ADC B	4
137	211	89	ADC C	4
138	212	8A	ADC D	4
139	213	8B	ADC E	4
140	214	8C	ADC H	4
141	215	8D	ADC L	4
142	216	8E	ADC M	7
143	217	8F	ADC A	4
144	220	90	SUB B	4
145	221	91	SUB C	4
146	222	92	SUB D	4
147	223	93	SUB E	4
148	224	94	SUB H	4
149	225	95	SUB L	4
150	226	96	SUB M	7
151	227	97	SUB A	4
152	230	98	SBB B	4
153	231	99	SBB C	4
154	232	9A	SBB D	4
155	233	9B	SBB E	4
156	234	9C	SBB H	4
157	235	9D	SBB L	4
158	236	9E	SBB M	7
159	237	9F	SBB A	4
160	240	A0	ANA B	4
161	241	A1	ANA C	4
162	242	A2	ANA D	4
163	243	A3	ANA E	4
164	244	A4	ANA H	4
165	245	A5	ANA L	4
166	246	A6	ANA M	7
167	247	A7	ANA A	4
168	250	A8	XRA B	4
169	251	A9	XRA C	4
170	252	AA	XRA D	4
171	253	AB	XRA E	4
172	254	AC	XRA H	4
173	255	AD	XRA L	4
174	256	AE	XRA M	7
175	257	AF	XRA A	4
176	260	B0	ORA B	4
177	261	B1	ORA C	4
178	262	B2	ORA D	4
179	263	B3	ORA E	4
180	264	B4	ORA H	4
181	265	B5	ORA L	4

DECIMAL	OCTAL	HEXADECIMAL	INSTRUCCION	NUMERO PERIODOS
182	266	B6	ORA M	7
183	267	B7	ORA A	4
184	270	B8	CMP B	4
185	271	B9	CMP C	4
186	272	BA	CMP D	4
187	273	BB	CMP E	4
188	274	BC	CMP H	4
189	275	BD	CMP L	4
190	276	BE	CMP M	7
191	277	BF	CMP A	4
192	300	C0	RNZ	5/11
193	301	C1	POP B	10
194	302	C2	JNZ	10
195	303	C3	JMP	10
196	304	C4	CNZ	11/17
197	305	C5	PUSH B	11
198	306	C6	ADI	7
199	307	C7	RST O	11
200	310	C8	RZ	5/11
201	311	C9	RET	10
202	312	CA	JZ	10
203	313	CB	—	—
204	314	CC	CZ	11/17
205	315	CD	CALL	17
206	316	CE	ACI	7
207	317	CF	RST B	11
208	320	D0	RNZ	5/11
209	321	D1	POP D	10
210	322	D2	JNC	10
211	323	D3	OUT	10
212	324	D4	CNC	11/17
213	325	D5	PUSH D	11
214	326	D6	SUI	7
215	327	D7	RST 10	11
216	330	D8	RC	5/11
217	331	D9	—	—
218	332	DA	JC	10
219	333	DB	IN	10
220	334	DC	CC	11/17
221	335	DD	—	—
222	336	DE	SBI	7
223	337	DF	RST 18	11
224	340	E0	RPO	5/11
225	341	E1	POP H	10
226	342	E2	JPO	10
227	343	E3	XTHL	18
228	344	E4	CPO	11/17
229	345	E5	PUSH H	11
230	346	E6	ANI	7
231	347	E7	RST 20	11

DECIMAL	OCTAL	HEXADECIMAL	INSTRUCCION	NUMERO PERIODOS
232	350	E8	RPE	5/11
233	351	E9	PCHL	5
234	352	EA	JPE	10
235	353	EB	XCHG	4
236	354	EC	CPE	11/17
237	355	ED	—	—
238	356	EE	XRI	7
239	357	EF	RST 28	11
240	360	F0	RP	5/11
241	361	F1	POP A	10
242	362	F2	JP	10
243	363	F3	DI	4
244	364	F4	CP	11/17
245	365	F5	PUSH A	11
246	366	F6	ORI	7
247	367	F7	RST 30	11
248	370	F8	RM	5/11
249	371	F9	SPHL	5
250	372	FA	JM	10
251	373	FB	EI	4
252	374	FC	CM	11/17
253	375	FD	—	—
254	376	FE	CPI	7
255	377	FF	RST 38	11

LENGUAJE ENSAMBLADOR, APLICADO AL MICROPROCESADOR INTEL 8080

Se exponen a continuación las normas y características del lenguaje Ensamblador, propuestas por el mismo fabricante: Intel. El empleo de este lenguaje para otro tipo de microprocesador es muy similar.

Cómo se utiliza el lenguaje Ensamblador

Al examinar el contenido de la memoria de un computador, un programa aparece como una serie de dígitos hexadecimales, que la CPU interpreta como códigos de instrucción, direcciones o datos. Sería posible escribir un programa en esta forma, pero resultaría un proceso lento y costoso.

Veamos un programa tal como se ve reflejado en la memoria:

Dirección de memoria hexadecimal	Contenido
1432	7E
1433	C3
1434	C4
1435	14
1436	
.	
.	
14C3	FF
14C4	2E
14C5	36
14C6	77

Supongamos que los registros H y L contienen respectivamente 14_{Hex} y $C3_{Hex}$. El programa opera como sigue:

El Byte de 1432 especifica que el acumulador se carga con el contenido del byte de memoria 14C3.

Los bytes del 1433 hasta 1435 indican que la ejecución del programa debe continuar en la posición 14C4.

Los bytes 14C4 y 14C5 especifican que el registro L se carga con el número 36.

El byte 14C6 indica que el contenido del acumulador se almacena en la posición de memoria 1436.

Supongamos ahora que, a causa de haber descubierto un error en el programa, es necesario añadir una instrucción en la posición siguiente a 1432. El programa debería modificarse en esta forma:

Dirección de memoria hexadecimal	Código exterior	Nuevo código
1432	7E	7E
1433	C3	Nueva instrucción
1434	C4	C3
1435	14	C5
1436		14
1437		
14C3	FF	
14C4	2E	FF
14C5	36	2E
14C6	77	37
14C7		77

Podrían añadirse más instrucciones y efectuar más cambios, pero cada vez el riesgo de error sería mayor, agravado además por la ilegibilidad del programa.

Para no tener que recurrir a esta forma engorrosa de trabajar, sobre todo en programas de cierta complejidad, el primer paso consiste en utilizar el lenguaje Ensamblador, que proporciona una notación de las instrucciones completamente legible, y evita al programador el tener que referirse a direcciones de memoria específicas.

El lenguaje Ensamblador es, en síntesis, una secuencia de instrucciones, que se convierten a un código hexadecimal ejecutable por la máquina a través de un programa llamado ASSEMBLER o ENSAMBLADOR.

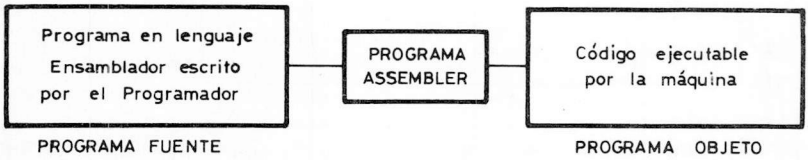


Fig. 6-11.–Conversión del "programa fuente", en lenguaje Ensamblador, en "programa objeto" mediante el ASSEMBLER.

Tal como se ilustra en la figura 6-11, el programa en lenguaje Ensamblador generado por el programador, se denomina PROGRAMA FUENTE. El programa ASSEMBLER convierte el PROGRAMA FUENTE en su equivalente en hexadecimal, denominado PROGRAMA OBJETO.

Por ejemplo:

Programa fuente		**Una posible versión de programa objeto**
BIEN: MOV A, B		78
CPI 'C'		FE43
JZ MAL	→ se convierte por el Assembler en →	CA7C3D
.		
.		
.		
MAL: MOV M, A		77

Nota: Tanto en éste como en posteriores ejemplos, no es necesario descifrar las operaciones que definen las instrucciones. Con ellos única-

mente se pretende mostrar algunas formas de operar típicas del lenguaje Ensamblador. El significado de las instrucciones individualmente se discutirá posteriormente.

Si ahora hay que insertar una nueva instrucción, únicamente se requiere realizar un cambio.

```
BIEN:   MOV    A, B
(Nuevas instrucción insertada aquí)
        CPI    'C'
        JZ     MAL
        .
        .
        .
MAL:    MOV    M, A
```

El Assembler será el que coloque esta instrucción con su código en su lugar adecuado, decalando el resto del programa en la memoria.

Sintaxis

Las instrucciones del lenguaje Ensamblador deben atenerse a una serie de reglas, tal como se describe en este apartado. Dentro de una instrucción hay cuatro partes o campos separados:

El campo 1 es el campo de la ETIQUETA, que es el nombre utilizado para referenciar la dirección de una instrucción.

El campo 2 es el del CODIGO. En él se especifica la operación a ejecutar.

El campo 3 corresponde al campo del OPERANDO. En él se definen la dirección de memoria o el dato precisado por el código.

Por último, el campo 4 corresponde al de los COMENTARIOS. Se pone a criterio del programador, y es ignorado por el Assembler. Su utilidad se reduce a servir como guía al programador, definiendo las operaciones que se realizan, lo que hace más comprensible el programa.

Estos campos deben estar separados por espacios en blanco, aún cuando su número no esté sujeto a norma alguna.

Antes de describir al detalle cada uno de los campos, veamos algunos ejemplos generales:

Etiqueta	Código	Operando	Comentarios
AQUI:	MVI	C, O	; Cargar O en el registro C
ALLI:	DB	3AH	; Creación un byte de datos ; constante
LAZO:	ADD	E	; Sumar el contenido del re- ; gistro E al acumulador
	RCL		; Desplazar el acumulador a la izquierda

Campo de la Etiqueta

Es un campo de utilización opcional que, cuando está presente, puede tener una longitud de 1 a 5 caracteres. El primer carácter de la etiqueta debe ser una letra del alfabeto o uno de los caracteres especiales @ o ?. Estos caracteres deben ir seguidos de dos puntos (:). Los códigos de instrucción, nombres de pseudo-instrucciones y nombres de registros están especialmente definidos por el Assembler y no pueden ser utilizados como etiquetas.

He aquí algunos ejemplos de etiquetas válidas:

LABEL:
F14F:
@AQUI:
? CERO:

Los siguientes ejemplos muestran etiquetas no válidas:

123:	comienza con un código decimal
LABEL	no va seguida de los dos puntos
ADD:	es un código de una instrucción
END:	es una pseudo-instrucción

En el caso de que una etiqueta tenga más de cinco caracteres, únicamente se reconocerán por el Assambler los cinco primeros:

INSTRUCCION: se leerá como INSTR:

Una etiqueta define solamente una dirección. No puede repetirse:

```
AQUI:  JMP    ALLI
       ...
       ...
ALLI:  MOV    C, D
       ...
       ...
ALLI:  CALL   SUB
```

no es correcto. El Assembler no puede determinar qué dirección es a la que debe ir la instrucción JMP.

No obstante, la siguiente secuencia de instrucciones es válida:

```
LAZ01:                      ; Primera etiqueta
LAZ02: MOV    C, D,         ; Segunda etiqueta
       ...
       JMP    LAZ01
       ...
       JMP    LAZ02
```

Ambas instrucciones JMP provocarán un salto de programa a la instrucción MOV.

Campo del Código

Este campo contiene un código que define la operación a realizar por la máquina (suma, resta, salto, etc.). Cada una de las instrucciones, ya descritas, tiene un código determinado, que es privativo de la misma, y que debe aparecer en el campo del código. Por ejemplo, las letras "JMP" definen exclusivamente la operación de "salto incondicional".

Después de las letras del campo del código debe haber, como mínimo, un espacio en blanco:

```
AQUI: JMP    ALLI
```

es válido, pero:

```
AQUI: JMPALLI
```

no lo es.

Campo del operando

La información contenida en este campo se usa conjuntamente con el campo del código, a fin de definir con precisión la operación a ejecutar por la instrucción. Según el contenido del campo del código, el campo del operando puede no existir, o consistir en una cifra o palabra, o bien en dos, separadas ambas por una coma.

Hay cuatro tipos de información válidos como campo del operando, lo que da lugar a nueva forma de expresarlo, según la figura 6-12.

INFORMACION DEL CAMPO DEL OPERANDO	
Información requerida	**Formas de especificarla**
a) Registro	1) Dato hexadecimal
b) Par de registros	2) Dato decimal
c) Dato inmediato	3) Dato octal
d) Dirección de 16 bits	4) Dato binario
	5) Contador de programa (S)
	6) Constante ASCII
	7) Etiquetas a las que se ha asignado valores
	8) Etiquetas de instrucción
	9) Expresiones

Fig. 6-12.–Informe del campo del operando.

Estas nueve formas de especificar el operando son las siguientes:

1) *Dato hexadecimal.* Todo número hexadecimal debe ir seguido de la letra *H*, y debe comenzar con un dígito decimal (0 a 9).

Ejemplo:

Etiqueta	Código	Operando	Comentarios
AQUI:	MVI	C, *O*BA*H*	; Cargar en el registro C ; el número hexadecimal BA

2) *Dato decimal.* Los números decimales pueden ir seguidos de la letra D, *o bien ir solos.*

Ejemplo:

Etiqueta	Código	Operando	Comentarios
ABC:	MVI	E, 105	; Cargar en el registro E ; 105

3) *Dato octal.* Todo número octal debe ir seguido de una de las letras *"O"* ó "Q".

Ejemplo:

Etiqueta	Código	Operando	Comentarios
LABEL:	MVI	A, 72*O*	; Cargar en el acumulador ; el número octal 72

4) *Dato binario.* Todo número binario debe ir seguido de la letra "B".

Ejemplo:

Etiqueta	Código	Operando	Comentarios
AHORA:	MVI	10B, 11110110*B*	; Cargar en el registro ; dos (registro D) OF6H
SALTO:	JMP	0010111011111010B	; Saltar a la dirección de ; memoria ; 2EFA

5) El contenido actual del contador de programa. Este se define con el carácter S y equivale a la dirección de la instrucción en ejecución.

Ejemplo:

Etiqueta	Código	Operando
IR:	JMP	S + 6

Esta instrucción provoca un salto en el programa a la dirección de memoria, 6 unidades superior a la que contiene la instrucción JMP.

6) Una constante ASCII. Son uno o más caracteres ASCII, encerrados entre comillas.

Ejemplo:

Etiqueta	Código	Operando	Comentarios
CARAC:	MVI	E, "."	; Cargar en el registro E los ; ocho bits de representación ; del carácter ASCII "."

7) Etiquetas a las que se les ha asignado un valor numérico por el Assembler, como por ejemplo las siguientes:

B	asignado	a	0	representa	el registro B
C	"	"	1	"	" C
D	"	"	2	"	" D
E	"	"	3	"	" E
H	"	"	4	"	" H
L	"	"	5	"	" L
M	"	"	6	"	una referencia a memoria
A	"	"	7	"	el registro acumulador

Ejemplo:

Supongamos que VALOR equivale al número hexadecimal 9F. Las siguientes instrucciones cargan 9F en el registro D:

Etiqueta	Código	Operando
A1:	MVI	D, VALOR
A2:	MVI	2, 9FH
A3:	MVI	2, VALOR

8) Etiquetas que aparecen en el campo de la etiqueta de otra instrucción.

Ejemplo:

Etiqueta	Código	Operando	Comentarios
AQUI:	JMP	ALLI	; Salto a la instrucción de ; ALLI
	...		
	...		
ALLI:	MVI	D, 9FH	

9) Las expresiones lógicas y aritméticas, descritas anteriormente en los apartados de 1) a 8), se representan mediate los operadores + (suma), – (resta o cambio de signo), X (multiplicación), / (división), MOD (módulo), los operadores lógicos NOT, AND, OR, XOR, SXR (rotación hacia la derecha) y SHL (rotación hacia la izquierda).

Todos los operadores tratan sus argumentos como cantidades de 15 bits, y generan como resultados cantidades de 16 bits.

El operador + genera la suma aritmética de sus operandos.

El operador – genera la resta aritmética de sus operandos, cuando se usa como sustracción, o como aritmética negativa, cuando se utiliza como cambio de signo.

El operador X indica el producto aritmético de los dos operandos.

El operador / provoca el cociente entre los dos operandos, descartando el resto.

El operador MOD da lugar a la obtención del resto, después de dividir el primer operando entre el segundo.

El operador NOT realiza el complemento de cada bit del operando.

El operador AND realiza la operación lógica AND bit a bit entre los operandos.

El operador OR realiza la operación lógica OR bit a bit entre los operandos.

El operador XOR realiza la operación lógica O-EXCLUSIVO bit a bit entre los operandos.

Los operandos SHR y SHL realiza un decalaje del primer operando a derecha e izquierda respectivamente el número de posiciones definidas por el segundo operando, colocando ceros en las posiciones decaladas.

El programador debe asegurarse de que el resultado generado por una de estas operaciones cumple los requisitos necesarios. Por ejemplo, el segundo operando de una instrucción MVI debe ser una cantidad de 8 bits.

Así, la instrucción

```
MVI     H, NOT O
```

no es válida, puesto que la operación NOT O genera el número hexadecimal FFFF.

Sin embargo, la instrucción

```
MVI     H, NOT O AND OFFH
```

es válida, pues los ocho bits más significativos del resultado se ponen a cero, obteniéndose un resultado total de 8 bits.

Nota: Una instrucción puesta entre paréntesis es una expresión válida, siendo un campo opcional. Su valor es el del código hexadecimal de la instrucción.

Ejemplos:

Etiqueta	Código	Operando	Direc. Memoria arbitraria
AQUI:	MVI	C, AQUI SHR 8	2EJA

Esta instrucción carga el número hexadecimal 2E (la dirección de 16 bits de AQUI decalada 8 bits hacia la derecha) en el registro C.

Etiqueta	Código	Operando
PROX:	MVI	D, 34 + 40H/2

Esta instrucción cargará la cantidad 34 + (64/2) = 34 + 32 = 66, en el registro D.

Etiqueta	Código	Operando
INS:	DB	(ADD C)

Mediante esta instrucción se define un byte de un valor de 81H (código hexadecimal de la instrucción ADD C) en la posición INS.

Las expresiones producidas por los operadores deben ser evaluadas en el siguiente orden de prioridad:

1. Expresiones entre paréntesis
2. *, /, MOD, SHL, SHR
3. +, –
4. NOT
5. AND
6. OR, XOR

En el caso de las expresiones entre paréntesis, el contenido entre los mismos debe evaluarse primero.

Ejemplo:

La instrucción

MVI D, $(34 + 40_H)/2$

cargará en el registro D el valor

$$(34 + 64)/2 = 98/2 = 49$$

Los operadores MOD, SHL, NOT, AND, OR y XOR deben estar separados de sus operandos como mínimo por un espacio en blanco. Así, la instrucción

MVI C, VALOR ANDOFH

no es válida.

En el caso de las nueve anteriores especificaciones de datos del operando, puede utilizarse, asimismo, los siguientes cuatro tipos de información:

a) Un registro (o código indicativo de una referencia a memoria), definido como fuente o destino de datos en una operación descrita en los apartados 1, 2, 3, 4, 7 ó 9, puede usarse para especificar el registro o referencia a memoria mencionado, pero el número finalmente evaluado debe estar entre 0 y 7.

VALOR	REGISTRO
0	B
1	C
2	D
3	E
4	H
5	L
6	Referencia a memoria
7	Acumulador

Ejemplo:

Etiqueta	Código	Operando
INS1:	MVI	REG4, 2EH
INS2:	MVI	4H, 2EH
INS3:	MVI	8/2, 2EH

Suponiendo que REG4 ha sido equiparado a 4, todas las instrucciones anteriores cargarán el valor 2EH en el registro H.

b) Un par de registros utilizado como fuente o destino de una operación con datos. Los pares de registros se especifican como sigue:

Especificación	Par de registros
B	Registros B y C
C	Registros D y E
H	Registros H y L
PSW	Un byte que indica el estado de los bits de condición y el registro A
SP	El registro de 16 bits de puntero de stack.

Nota: El valor binario que representa cada par de registros varía de una a otra instrucción. No obstante, el programador debe asignarles siempre su designación alfabética.

Ejemplo:

Etiqueta	Código	Operando	Comentarios
	PUSH	D	; Cargar el contenido de los ; registro D y E en el stack
	INX	SP	; Incrementar el número de ; 16 bits del puntero de stack

Nota: Para una mayor compresión y ampliación del tema se recomienda al lector acudir a la información específica de los fabricantes de microprocesadores.

APENDICE

INSTRUCCIONES DEL MICROPROCESADOR SIGNETICS 2650

INSTRUCCIONES

Las instrucciones del microprocesador 2650 se pueden clasificar de acuerdo con varios criterios.

Clasificación de las instrucciones de acuerdo con su longitud

El microprocesador 2650 utiliza instrucciones de longitud variable.

La longitud de la instrucción depende del tipo de operación a realizar y de la forma de obtener la dirección del operando.

a) Instrucciones de un octeto

Se incluyen dentro de este grupo las instrucciones que operan exclusivamente con los registros de la unidad central de proceso R_0 a R_6 y otras instrucciones de control y entrada/salida: Operación nula (*No operation*), Paro (*Halt*), Entrada (*Read*) y Salida (*Write*) de información desde o hacia un periférico, etc.

b) Instrucciones de dos octetos

Pertenecen a este grupo todas las instrucciones que operan con un dato almacenado en memoria y utilizan direccionamiento relativo e inmediato.

c) Instrucciones de tres octetos

Pertenecen a este grupo aquellas instrucciones que operan con un dato contenido en memoria y utilizan los tipos de direccionamiento absoluto e indexado.

Clasificación de las instrucciones de acuerdo con el código de operación

1.º) Instrucciones aritméticas y lógicas y de transferencia o movimiento de datos entre registros.

2.º) Instrucciones aritméticas y lógicas, y de transferencia entre un registro del microprocesador y una posición de memoria.

3.°) Instrucciones de salto o bifurcación condicional e incondicional.

4.°) Instrucciones de control.

5.°) Instrucciones de entrada y salida.

El código de operación de la instrucción está siempre contenido en el primer octeto que se lee de la memoria en un ciclo de búsqueda de instrucción. Este octeto está dividido en tres campos tal como indica la figura 6-13.

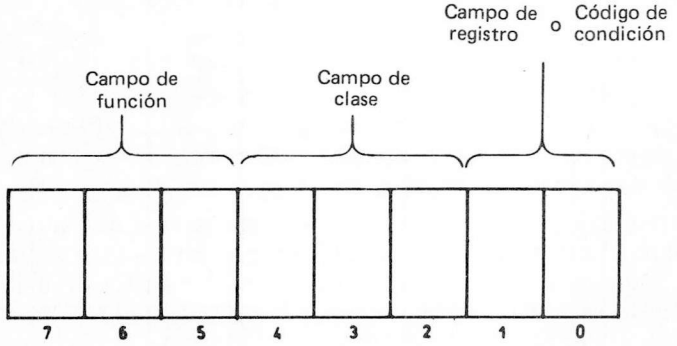


Fig. 6-13.–Diferentes campos de un octeto.

El campo de Clase especifica el grupo al cual pertenece la instrucción entre los definidos en párrafos anteriores, el tipo de direccionamiento y el número de ciclos de la unidad de control de microprocesador necesarios para cada instrucción. El campo de Clase especifica también con una sola excepción, el número de octetos que ocupa la instrucción de la memoria.

En los grupos de instrucciones aritméticas, el campo de función especifica una de las ocho operaciones siguientes:

Cargar (Load)
O exclusiva (Exclusive OR)
Y lógica (AND)
O lógica (Inclusive OR)
Suma (Add)
Resta (Subtract)
Almacenar (Store)
Comparación (Compare)

En el grupo de las instrucciones de bifurcación, el campo de función especifica una de las ocho operaciones siguientes:

- Bifurcar si se cumple la condición.
- Bifurcar a subrutina si se cumple la condición.
- Bifurcar si el contenido del registro seleccionado no es nulo.
- Bifurcar a subrutina si el contenido del registro seleccionado no es nulo.
- Bifurcar si no se cumple la condición.
- Bifurcar a un subrutina si no se cumple la condición.
- Incrementar el contenido de un registro y bifurcar si su contenido no es nulo.
- Disminuir en uno el contenido de un registro y bifurcar si su contenido no es nulo.

En los grupos de instrucciones de control, Entrada/Salida y rotación, ambos campos, el de Clase y el de Función especifican conjuntamente el tipo de acción a realizar por la instrucción.

El campo de registro o campo de control tiene significado diferente según la clase a la que pertenece la instrucción. En las instrucciones aritméticas, de transferencia y de entrada-salida se especifica el registro con cuyo contenido se realiza la operación, el que recibe el resultado o el registro de índice si el modo de direccionamiento es indexado, o un código de condición en las instrucciones de bifurcación condicional.

En las instrucciones de operación entre registros y en las de modo de direccionamiento indexado, el registro cero está direccionado de una forma implícita y según el tipo de operación proporciona uno de los operandos o recibe el resultado. Todas las demás instrucciones que operan con el contenido de un registro, especifican el registro utilizado mediante el campo de registro, excepto las instrucciones de bifurcación indexada en las que se especifica el registro R_3.

Las instrucciones de bifurcación condicional utilizan el campo de registro como un código de condición, y lo comparan con el contenido del Registro del código de condición (CC) del Registro de estado y control del programa para tomar la decisión de bifurcar según el resultado de dicha comparación. Algunas instrucciones utilizan el campo de registro como parte del código de operación y por lo tanto no permiten seleccionar el registro a utilizar.

TIPOS DE DIRECCIONAMIENTO

Se entiende por direccionamiento la forma en que una unidad central de proceso obtiene la dirección del operando.

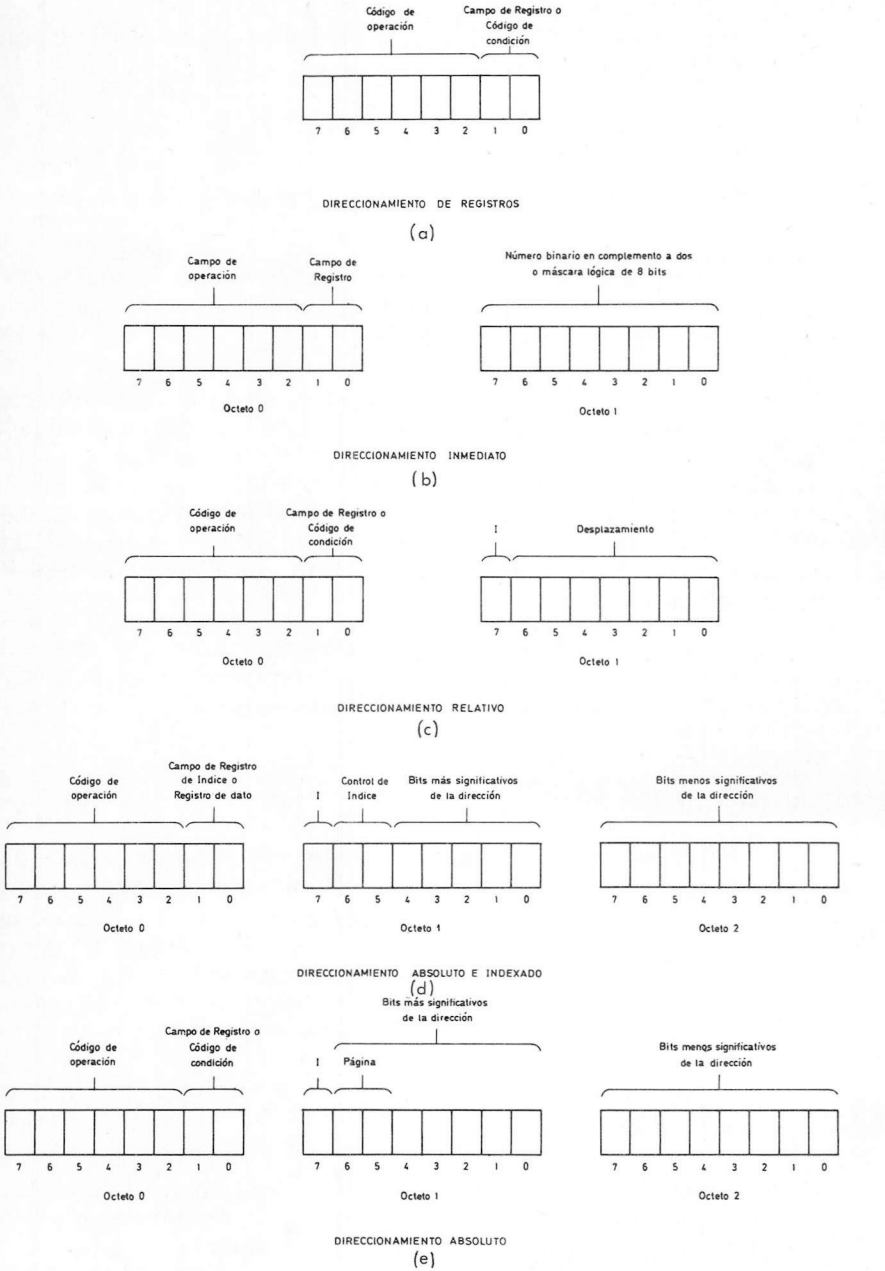


Fig. 6-14.–Diferentes modos de direccionamiento del microprocesador 2650.

El microprocesador 2650 utiliza los siguientes tipos de direccionamiento (ver la figura 6-14).

- De registros.
- Inmediato.
- Relativo.
- Indexado.
- Absoluto.
- Indirecto.

Direccionamiento de registros

Todas las instrucciones que realizan operaciones entre registros tienen una longitud de un octeto. El formato de este tipo de direccionamiento se indica e la figura 6-14 a).

Dado que solamente se utilizan dos bits para seleccionar un registro, el registro R_0 contiene siempre uno de los operandos y mediante los citados bits se selecciona el registro que suministra el otro operando de uno de los dos grupos (el grupo lo selecciona el bit correspondiente del Registro de estado del programa).

El registro R_0 se puede también seleccionar de forma explícita en las instrucciones que utilizan un solo operando, como por ejemplo la transferencia de información de una posición de memoria a un registro.

Direccionamiento inmediato

El dato con el cual se ha de realizar la operación indicada en el código de operación de la instrucción está contenido en la dirección de memoria siguiente a aquella en la cual se encuentra la instrucción. Según el código de operación contenido en la instrucción, dicho dato puede ser un número binario negativo o positivo o una máscara lógica. En la figura 6-14b) se representa el formato de una instrucción de direccionamiento inmediato.

Direccionamiento relativo

Todas las instrucciones que utilizan el direccionamiento relativo ocupan dos posiciones (dos octetos) de la memoria y utilizan un operando situado en memoria. El otro operando es el contenido de un registro, especificado mediante el Campo de Registro de la instrucción. El formato de

las instrucciones que utilizan el direccionamiento relativo se indica en la figura 6-14c).

Direccionamiento absoluto e indexado

El direccionamiento indexado se utiliza siempre en combinación con el absoluto, y exclusivamente en las instrucciones operativas y de transferencia que utilizan un operando situado en memoria (estas instrucciones no toman decisión de bifurcación). Las instrucciones que toman la decisión de bifurcar utilizan el direccionamiento absoluto.

Direccionamiento indexado y absoluto de las instrucciones operativas y de transferencia con la memoria

Las instrucciones que utilizan estos tipos de direccionamiento tienen una longitud de tres octetos (figura 6-14d).

Direccionamiento absoluto de las instrucciones de bifurcación

Las instrucciones de bifurcación que utilizan el direccionamiento absoluto son de tres octetos y poseen un formato como el especificado en la figura 6-14e).

Los bits 5 y 6 del octeto 1 se consideran los bits más significativos de la dirección y no se interpretan como bits de Control de Indice. Por lo tanto todas las instrucciones de bifurcación que poseen este formato no permiten la realización de direccionamiento indexado.

Existe una instrucción especial de bifurcación incondicional y otra de bifurcación incondicional a una subrutina, que realizan direccionamiento indexado utilizando como registro de índice R_3. Estas instrucciones se denominan mediante los códigos simbólicos BXA (*Branch indexed and absolute*) y BSXA (*Branch to subroutine indexed and absolute*).

Direccionamiento indirecto

El direccionamiento indirecto se utiliza en combinación con el relativo, el indexado y el absoluto. Al realizar direccionamiento indirecto, el microprocesador interpreta el contenido de la dirección como una nueva dirección en lugar de como un dato.

Cuando se combina el direccionamiento indirecto con el indexado primero se realiza aquél y después éste, lo cual suele denominarse postindexado.

La realización del direccionamiento indexado añade dos ciclos (6 períodos del generador de impulsos) al tiempo de ejecución normal de una instrucción. Las direcciones indirectas están constituidas por 15 bits almacenados en dos posiciones consecutivas de memoria y por lo tanto permiten direccionar cualquier posición de los 32K de memoria. Solamente se puede realizar un nivel de direccionamiento indirecto.

CODIGOS NEMOTECNICOS DE LAS INSTRUCCIONES EN LENGUAJE ENSAMBLADOR

Se indican seguidamente los códigos nemotécnicos de todas las instrucciones del microprocesador 2650.

El significado de los símbolos utilizados es el siguiente:

r indicativo de un registro que ha de convertirse en lenguaje máquina en un número comprendido entre 0 y 3.
v indicativo de un valor numérico.
* indicativo de direccionamiento indirecto.
a indicativo de una dirección.
x indicativo de un registro de índice.
X indicativo de un registro de índice con opción de autoincremento o autodecremento.

Es interesante tener en cuenta los siguientes puntos:

1.°) El uso del indicativo de direccionamiento indirecto es opcional.

2.°) Cuando se especifica un registro de índice, puede ir seguido por ",+" o ",–" que indica que se autoincrementa o autodecrementa. Por ejemplo:

LODA, 0

DPR, R3, +

Las instrucciones BXA y BSXA son excepciones y no permiten la realización de autoincremento o autodecremento.

3.°) Una instrucción que hace referencia a memoria que requiere direccionamiento indexado solamente puede utilizar el registro R_0 para recibir el operando.

4.°) Aunque se especifique una expresión en las instrucciones que utilizan direccionamiento relativo, el traductor la convierte en un valor numérico V ($-64 \leqslant V \leqslant +63$).

5.°) Si se utiliza un indicativo de registro de índice en las instrucciones BXA o BSXA, ha de especificar el Registro R_3 (de uno de los

dos grupos). Cualquier otro valor en el campo del registro de índice producirá un error en el ensamblado. Sin embargo, no es necesario utilizar un indicativo de registro de índice en estas instrucciones. En este caso el traductor asigna directamente el registro R_3.

		Longitud (octetos)
Instrucciones de Carga/Almacenamiento		
LODZ r	Cargar registro R_0	1
LODI,r v	Cargar R_r inmediato	2
LODR,r (*) a	Cargar R_r con dir. relativo	2
LODA,r (*) a (,X)	Cargar R_r con dir. absoluto e indexado	3
STRZ r	Almacenar registro R_0	1
STRR,r (*) a	Almacenar R_r con dir. relativo	2
STRA,r (*) a (,X)	Almacenar R_r con dir. absoluto e indexado	3
Instrucciones aritméticas		
ADDZ r	Sumar el registro R_0	1
ADDI,r v	Sumar inmediato	2
ADDR,r (*) a	Sumar con dir. relativo	2
ADDA,r (*) a (,X)	Sumar con dir. absoluto e indexado	3
SUBZ r	Restar del registro R_0	1
SUBI,r v	Restar inmediato	2
SUBR,r (*) a	Restar con dir. relativo	2
SUBA,r (*) a (,X)	Restar con dir. absoluto e indexado	3
Instrucciones lógicas		
ANDZ r	Y con el registro R_0	1
ANDI,r v	Y inmediato	2
ANDR,r (*) a	Y con dir. relativo	2
ANDA,r (*) a (,X)	Y con dir. absoluto e indexado	3
IORZ r	O con el registro R_0	1
IORI,r v	O inmediato	2
IORR,r (*) a	O con dir. relativo	2
IORA,r (*) a (,X)	O con dir. absoluto e indexado	3
EORZ r	O exclusiva con el registro R_0	1
IORI,r v	O exclusiva inmediato	2
EORR,r (*) a	O exclusiva con dir. relativo	2
EORA,r (*) a (,X)	O exclusiva con dir. absoluto e indexado	3
Instrucciones de comparación		
COMZ r	Comparar con registro R_0	1
COMI,r v	Comparar inmediato	2
COMR,r (*) a	Comparar con dir. relativo	2
COMA,r (*) a (,X)	Comparar con dir. absoluto e indexado	3

		Longitud (octetos)
Instrucciones de rotación		
RRR,r	Rotar un registro hacia la derecha	1
RRL,r	Rotar un registro hacia la izquierda	1
Instrucciones de bifurcación		
BCTR,v (*) a	Bifurcar si Condición cierta (dir. relativo)	2
BCFR,v (*) a	Bifurcar si Condición falsa (dir. relativo)	2
BCTA,v (*) a	Bifurcar si Condición cierta (dir. absoluto)	3
BCFA,v (*) a	Bifurcar si Condición falsa (dir. absoluto)	3
BRNR,r (*) a	Bifurcar si $R_r \neq 0$ (dir. relativo)	2
BRNA,r (*) a	Bifurcar si $R_r \neq 0$ (dir. absoluto)	3
BIRR,r (*) a	Bifurcar si $R_r \neq 0$ aumentando (dir. relativo)	2
BIRA,r (*) a	Bifurcar si $R_r \neq 0$ aumentando (dir. absoluto)	3
BDRR,r (*) a	Bifurcar si $R_r \neq 0$ disminuyendo (dir. relativo)	2
BDRA,r (*) a (,x)	Bifurcar si $R_r \neq 0$ disminuyendo (dir. absoluto)	3
BXA (*) a	Bifurcar incondicional (dir. indexado y absoluto	3
ZBRR (*) a	Bifurcación incondicional a la página cero	2
Instrucciones de bifurcación y retorno de subrutina		
BSTR,v (*) a	Bifurcar a subrutina si Condición cierta (dir. relativo)	2
BSFR,v (*) a	Bifurcar a subrutina si Condición falsa (dir. relativo)	2
BSTA,v (*) a	Bifurcar a subrutina si Condición cierta (dir. absoluto)	3
BSFA,v (*) a	Bifurcar a subrutina si Condición falsa (dir. absoluto)	3
BSNR,r (*) a	Bifurcar a subrutina si $R_r \neq 0$ (dir. relativo)	2
BSNA,r (*) a	Bifurcar a subrutina si $R_r \neq 0$ (dir. absoluto)	3
BSXA (*) a (,x)	Bifurcar a subrutina incondicionalmente (dir. indexado)	3
RETC,v	Retorno de subrutina condicional	1
RETE,v	Retorno de subrutina y desinhibición de Interrupción condicional	1
ZBSR (*) ,a	Bifurcación a subrutina en la página cero Incondicional	2

			Longitud (octetos)
Instrucciones del Registro de estado y control del programa			
LPSU		Cargar el Reg. estado programa (Superior)	1
LPSL		Cargar el Reg. estado programa (Inferior)	1
SPSU		Cargar el Reg. estado programa (Superior)	1
SPSL		Cargar el Reg. estado programa (Inferior)	1
CPSU		Borrar el Reg. estado programa (Superior) selectivamente	2
CPSL	v	Borar el Reg. estado programa (Inferior) selectivamente	2
PPSU	v	Poner a uno el Reg. estado programa (Superior) selectivamente	2
PPSL	v	Poner a uno el Reg. estado programa (Inferior) selectivamente	2
TPSU	v	Comprobar el Reg. estado programa (Superior) selectivamente	2
TPSL	v	Comprobar el Reg. estado programa (Inferior) selectivamente	2
Instrucciones de entrada/salida			
WRTD,r		Salida de dato	1
REDD,r		Entrada de dato	1
WRTC,r		Salida de control	1
REDC,r		Entrada de control	1
WRTE,r	v	Salida extendida	2
REDE,r	v	Entrada extendida	2
Instrucciones especiales			
HALT		Parar (pasar al estado de espera)	1
DAR,r		Registro de corrección decimal	1
TMI,r	v	Comprobar el contenido de R_r comparándolo con una máscara	2
NOP		No operar	1

Finalmente, se ofrecen los códigos OP y las características más destacables del microprocesador 2650. Téngase en cuenta que Z representa el modo de direccionado por registro, I: Inmediato, R: Relativo, A: Absoluto (No Branch), B: Absoluto (Branch) y E: Variado.

JUEGO DE INSTRUCCIONES DEL MICROPROCESADOR 2650

	DESCRIPTION OF OPERATION	FORMAT	MNEMONIC	OP CODE R or CC 0	1	2	3	BYTES	CYCLES	BIT FORMAT	PSW BITS AFFECTED CC	IDC	C	OVF	SP	II	F	NOTE
LOAD/STORE	Load register zero	2	LOD Z	00	01	02	03	1	2	Z	•							1
	Load immediate	5	LOD I	04	05	06	07	2	2	I	•							1
	Load relative	7	LOD R	08	09	0A	0B	2	3	R	•							1,6
	Load absolute	10	LOD A	0C	0D	0E	0F	3	4	A	•							6
	Store register zero	2	STR Z	—	C1	C2	C3	1	2	Z	•							1
	Store relative	7	STR R	C8	C9	CA	CB	2	3	R								6
	Store absolute	10	STR A	CC	CD	CE	CF	3	4	A								6
ARITHMETIC	Add to register zero w/wo carry	2	ADD Z	80	81	82	83	1	2	Z	•	•	•	•				1
	Add immediate w/wo carry	5	ADD I	84	85	86	87	2	2	I	•	•	•	•				1
	Add relative w/wo carry	7	ADD R	88	89	8A	8B	2	3	R	•	•	•	•				1,6
	Add absolute w/wo carry	10	ADD A	8C	8D	8E	8F	3	4	A	•	•	•	•				1,6
	Subtract from register zero w/wo borrow	2	SUB Z	A0	A1	A2	A3	1	2	Z	•	•	•	•				1
	Subtract immediate w/wo borrow	5	SUB I	A4	A5	A6	A7	2	2	I	•	•	•	•				1
	Subtract relative w/wo borrow	7	SUB R	A8	A9	AA	AB	2	3	R	•	•	•	•				1,6
	Subtract absolute w/wo borrow	10	SUB A	AC	AD	AE	AF	3	4	A	•	•	•	•				1,6
	Decimal adjust register	3	DAR	94	95	96	97	1	3	Z	•							1,10
LOGICAL	AND to register zero	2	AND Z	—	41	42	43	1	2	Z	•							1
	AND immediate	5	AND I	44	45	46	47	2	2	I	•							1
	AND relative	7	AND R	48	49	4A	4B	2	3	R	•							1,6
	AND absolute	10	AND A	4C	4D	4E	4F	3	4	A	•							1,6
	Inclusive-OR to register zero	2	IOR Z	60	61	62	63	1	2	Z	•							1
	Inclusive-OR immediate	5	IOR I	64	65	66	67	2	2	I	•							1
	Inclusive-OR relative	7	IOR R	68	69	6A	6B	2	3	R	•							1,6
	Inclusive-OR absolute	10	IOR A	6C	6D	6E	6F	3	4	A	•							1,6
	Exclusive-OR to register zero	2	EOR Z	20	21	22	23	1	2	Z	•							1
	Exclusive-OR immediate	5	EOR I	24	25	26	27	2	2	I	•							1
	Exclusive-OR relative	7	EOR R	28	29	2A	2B	2	3	R	•							1,6
	Exclusive-OR absolute	10	EOR A	2C	2D	2E	2F	3	4	A	•							1,6
ROTATE/COMPARE	Compare to register zero arithmetic/logical	2	COM Z	E0	E1	E2	E3	1	2	Z	•							2
	Compare immediate arithmetic/logical	5	COM I	E4	E5	E6	E7	2	2	I	•							3
	Compare relative arithmetic/logical	7	COM R	E8	E9	EA	EB	2	3	R	•							3,6
	Compare absolute aritmetic/logical	10	COM A	EC	ED	EE	EF	3	4	A	•							3,6
	Rotate register w/wo carry	3	RRR	50	51	52	53	1	2	Z	•	•	•	•				1
	Rotate register left w/wo carry	3	RRL	D0	D1	D2	D3	1	2	Z	•	•	•	•				1
BRANCH	Branch on condition true relative	7	BCT R	18	19	1A	1B	2	3	R								7,8
	Branch on condition true absolute	8	BCT A	1C	1D	1E	1F	3	3	B								7,8
	Branch on condition false relative	7	BCF R	98	99	9A	—	2	3	R								7
	Branch on condition false absolute	8	BCF A	9C	9D	9E	—	3	3	B								7
	Branch on register non-zero relative	7	BRN R	58	59	5A	5B	2	3	R								7,8
	Branch on register non-zero absolute	8	BRN A	5C	5D	5E	5F	3	3	B								7,8

JUEGO DE INSTRUCCIONES DEL MICROPROCESADOR 2650

	DESCRIPTION OF OPERATION	FORMAT	MNEMONIC	OP CODE R or CC 0	1	2	3	BYTES	CYCLES	BIT FORMAT	PSW BITS AFFECTED CC	IDC	C	OVF	SP	II	F	NOTE
BRANCH	Branch on incrementing register relative	7	BIR R	D8	D9	DA	DB	2	3	R								7,8
	Branch on incrementing register absolute	8	BIR A	DC	DD	DE	DF	3	3	B								7,8
	Branch on decrementing register relative	7	BDR R	F8	F9	FA	FB	2	3	R								7,8
	Branch on decrementing register absolute	8	BDR A	FC	FD	FE	FF	3	3	B								7,8
	Zero branch relative, unconditional	6	ZBRR	—	—	—	9B	2	3	ER								6
	Branch indexed absolute, unconditional	9	BXA	—	—	—	9F	3	3	EB								5,6
SUBROUTINE BRANCH/RETURN	Branch to subroutine on condition true, relative	7	BST R	38	39	3A	3B	2	3	R					•			7,8
	Branch to subroutine on condition true, absolute	8	BST A	3C	3D	3E	3F	3	3	B					•			7,8
	Branch to subroutine on condition false, relative	7	BSF R	B8	B9	BA	—	2	3	R					•			7
	Branch to subroutine on condition false, absolute	8	BSF A	BC	BD	BE	—	3	3	B					•			7
	Branch to subroutine on non-zero register, relative	7	BSN R	78	79	7A	7B	2	3	R					•			7,8
	Branch to subroutine on non-zero register, absolute	8	BSN A	7C	7D	7E	7F	3	3	B					•			7,8
	Zero branch to subroutine relative, unconditional	6	ZBSR	—	—	—	BB	2	3	ER								6
	Branch to subroutine, indexed, absolute unconditional	9	BSXA	—	—	—	BF	3	3	EB								5,6
	Return from subroutine, conditional	3	RET C	14	15	16	17	1	3	Z					•			8
	Return from subroutine and enable interrupt, conditional	3	RET E	34	35	36	37	1	3	Z					•	•		8
INPUT/OUTPUT	Write data	3	WRTD	F0	F1	F2	F3	1	2	Z								
	Read data	3	REDD	70	71	72	73	1	2	Z	•							1
	Write control	3	WRTC	B0	B1	B2	B3	1	2	Z								
	Read control	3	REDC	30	31	32	33	1	2	Z	•							1
	Write extended	5	WRTE	D4	D5	D6	D7	2	3	I								
	Read extended	5	REDE	54	55	56	57	2	3	I	•							1
MISC.	Halt, enter wait state	1	HALT	40	—	—	—	1	1	E								
	No operation	1	NOP	C0	—	—	—	1	1	E								
	Test under mask immediate	5	TMI	F4	F5	F6	F7	2	3	I	•							4
PROGRAM STATUS	Load program status, upper	1	LPS U	—	—	92	—	1	2	E					•	•	•	
	Load program status, lower	1	LPS L	—	—	93	—	1	2	E	•	•	•	•				9
	Store program status, upper	1	SPS U	—	—	12	—	1	2	E	•							1
	Store program status, lower	1	SPS L	—	—	13	—	1	2	E	•							1
	Clear program status, upper, masked	4	CPS U	—	—	74	—	2	3	EI					•	•	•	
	Clear program status, lower, masked	4	CPS L	—	—	75	—	2	3	EI	•	•	•	•				9
	Preset program status, upper, masked	4	PPS U	—	—	76	—	2	3	EI					•	•	•	
	Preset program status, lower, masked	4	PPS L	—	—	77	—	2	3	EI	•	•	•	•				9
	Test program status, upper, masked	4	TPS U	—	—	B4	—	2	3	EI	•							4
	Test program status, lower, masked	4	TPS L	—	—	B5	—	2	3	EI	•							4

NOTES

1. Condition code (CC1, CC0). 01 if positive, 00 if zero, 10 if negative.
2. Condition code (CC1, CC0): 01 if R0 > r, 00 if R0 = r, 10 if R0 < r.
3. Condition code (CC1, CC0): 01 if r > V, 00 if r = V, 10 if r < V.
4. Condition code (CC1, CC0) 00 if all salected bits are 1s, 10 if not all the selected bits are 1s.
5. Index register must be register 3 or 3'.
6. Requires two additional cycles indirection is specified.
7. Requires two additional cycles if indirection is specified and branch is taken.
8. Specify CC = 11 for unconditional branch.
9. RS, WC and COM bits in PSW are also affected.
10. CC assumes number in register is a binary number.

ASSEMBLER FORMATS

Formar No.	Format		Addressing Mode
1	OP		E
2	OP	r	Z
3	OP, r/c		Z
4	OP	v	Δ_I
5	OP, r	v	I
6	OP	[*] z	Δ_R
7	OP, r/c	[*] d	R
8	OP, r/c	[*] a	B
9	OP	[*] a ,3	Δ_B
10	OP, r	[*] p [,x] [,±]	A

Optional data enclosed in [ ]. Brackets are not part of syntax.
OP = operation code
r = register number (0-3)
c = condition code (0-3)
v = value (0-255)
* indicates indirect addressig
z = zero page addresses (o-63, 8128-8191)
d = relative displacement addresses (-64 to +63
a = addresses (0-32767)
p = page addresses (0-8191)
x = index register number (0-3)
+ = auto-increment
– = auto-decrement
Δ 8-bit operation code (no r/c field requied)

CAPITULO 7

Familias de circuitos integrados para microprocesadores. Periféricos

INTRODUCCION

El microprocesador no es operativo ni puede funcionar sin la colaboración y conexión a la memoria (de programa y de datos), a los módulos de entrada y salida y a un generador de la señal de reloj si no lo tiene incluido. Así, en la figura 7-1 se aprecia la interrelación de la CPU de la familia R 6500 con los restantes componentes del sistema microcomputador a través de los buses de direccionamiento, de datos y de control. La memoria que contiene el programa de instrucciones es la ROM, y la que maneja los datos variables y temporales es la RAM. También hay un módulo de entrada/salida (I/O), que transfiere datos en serie y en paralelo, y otro módulo más completo, que dispone además de una puerta I/O para transferencia de datos en paralelo de un circuito contador-temporizador o timer y de memoria interna.

La necesidad de combinar la CPU con otros circuitos integrados, compatibles con ella, para construir el sistema, ha impulsado a muchos fabricantes a ofrecer a los usuarios, junto con el microprocesador, toda una familia de chips compatibles, que cubran las necesidades de implementar cualquier aplicación del microprocesador; tal es el caso de los fabricantes de microprocesadores que se han estudiado.

Dentro de cada "familia microcomputador" de circuitos integrados, existen 5 tipos diferentes:

A) *Circuitos de memoria:* Tipos RAM, ROM, PROM y REPROM para contener los datos temporales el primero, y el programa de instrucciones los 3 restantes.

B) *Circuitos generadores de la señal de reloj:* Sincronizan y delimitan los tiempos de los ciclos de trabajo del microprocesador.

C) *Circuitos de entrada y salida:* Transfiere datos desde el exterior a la CPU para su procesamiento, o bien desde la CPU al exterior.

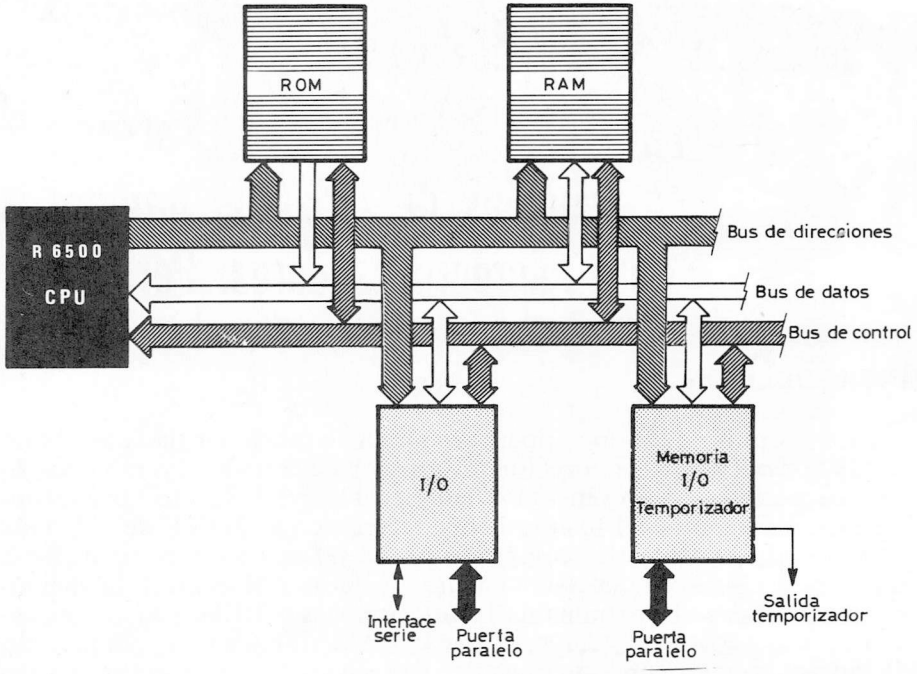


Fig. 7-1.–Interconexión de la CPU con los restantes componentes para formar un sistema microcomputador.

D) *Circuitos de aplicaciones especiales:* Tales como adaptadores y controladores de determinados periféricos (floppy disk, impresoras, circuitos DMA, que permiten el acceso directo desde el exterior a la memoria, etc.).

E) *Circuitos mixtos:* Que combinan varios de los tipos anteriores.

Se comienza describiendo la familia correspondiente a la firma Motorola, que fue una de las pioneras en introducir chips específicos para la constitución de sistemas microcomputadores.

1.ª PARTE: FAMILIA MICROCOMPUTADOR PARA EL MC 6800

CONJUNTO DE ELEMENTOS QUE COMPONEN LA FAMILIA

Además del elemento principal de la familia, que lógicamente es el microprocesador MC 6800, el resto de los circuitos integrados (más representativos) de la familia son los siguientes:

A) "Memorias": MCM 6830 (ROM), y MCM 6810 (RAM).

B) "Generador de señales de reloj": MC 6870.

C) "Módulos de entrada-salida": MC 6820 (Adaptador de datos en paralelo, PIA), MC 6850 (Módulo de entrada y salida de datos en serie ACIA).

D) "Circuitos especiales": MC 6828 (Controlador de interrupciones, PIC) y MC 6860 (Interface a modem de baja velocidad).

E) "Circuitos mixtos": MC 6846 (RAM, I/O, TIMER).

A través de los 3 buses de la CPU puede adaptarse al sistema cualquiera de los circuitos integrados que forman la familia, tal como se ha esquematizado gráficamente en la figura 7-2.

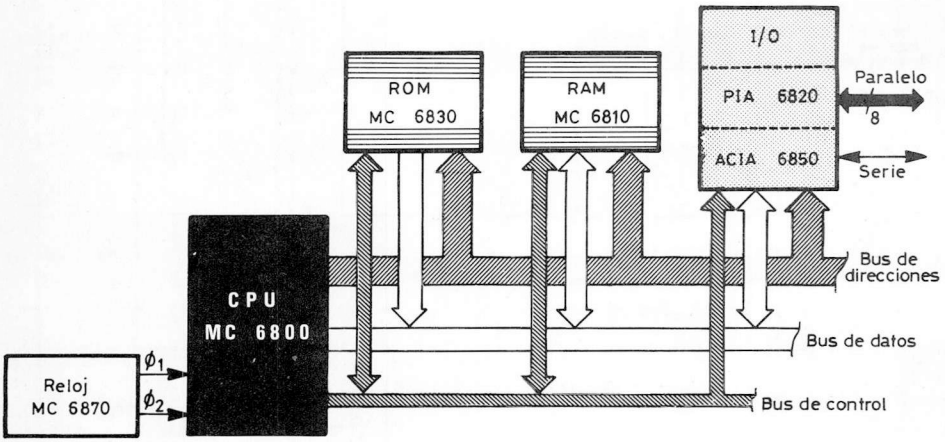


Fig. 7-2.–Interconexión de los elementos de la familia microcomputadora a los buses.

MEMORIA ROM MCM 6830

Se trata de una memoria de sólo lectura, tipo ROM, con capacidad para 1 K bytes (1.024 bytes) de 8 bits cada uno. Dispone de 10 líneas que se conectan al bus de direcciones, a través de las cuales se selecciona una posición de la memoria tras su correspondiente decodificación. La salida del contenido de la posición seleccionada se transfiere al bus de datos mediante 8 líneas de salida. Su tensión de alimentación es de +5 V. Finalmente dispone de 4 líneas para selección de chip, con las cuales se pueden controlar y decodificar hasta 16 chips MCM 6830, aunque, a consecuencia de utilizarse una de dichas líneas para introducir la señal de reloj ϕ_2, el número máximo de chips a controlar será sólo de 8. (Ver la figura 7-3.)

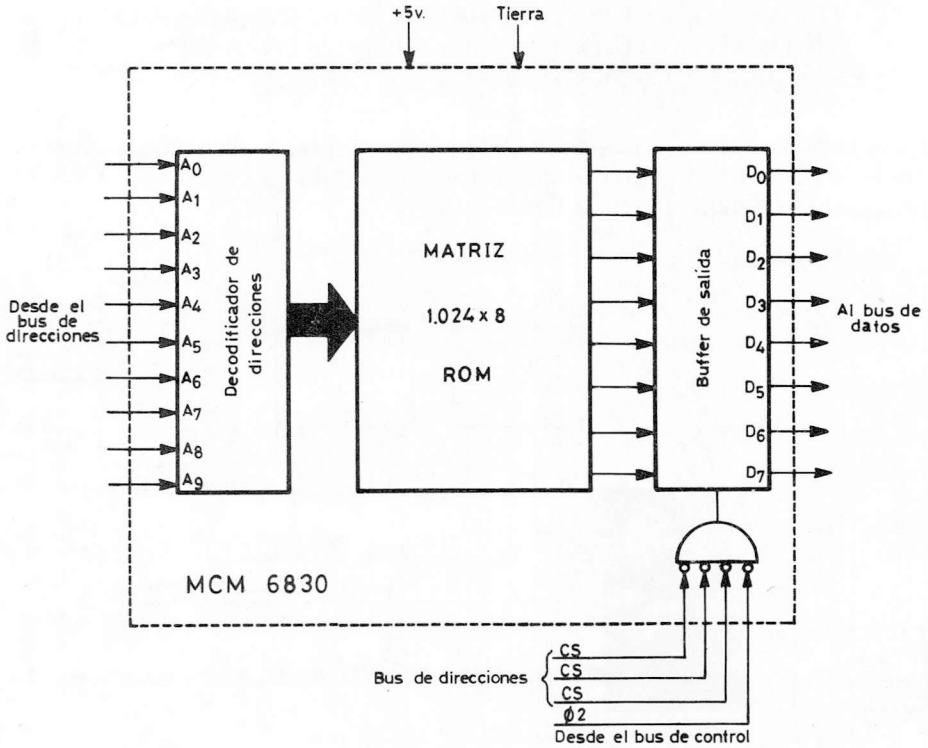


Fig. 7-3.–Estructura de la memoria ROM, MCM 6830.

La memoria tiene salida de datos en triestado, es compatible con niveles TTL y el tiempo de acceso máximo es de 500 ns.

Existe la versión MCM 6832 de características similares, pero con una capacidad de 2 K X 8 bits.

MEMORIA RAM MCM 6810

Se trata de una memoria viva, de lectura y escritura, tipo RAM, con capacidad para 128 X 8 bits. Para seleccionar una posición de su matriz dispone de 7 líneas, que reciben la información desde el bus de direccionamiento, según se muestra en la figura 7-4. Existen 8 pines dedicados a la entrada y salida de datos (escritura y lectura), que se conectan al bus de datos. Para poder manejar varios chips de este tipo, cuando con uno sólo no se obtiene la suficiente capacidad de memoria, el MCM 6810 posee 4 líneas de selección de chip, además de la que se emplea para introducir la señal de reloj ϕ_2 y otra que introduce +5 V, quedando otras cuatro negadas ($\overline{CS}$: activas con nivel bajo), que pueden controlar y seleccionar hasta un máximo de 16 chips.

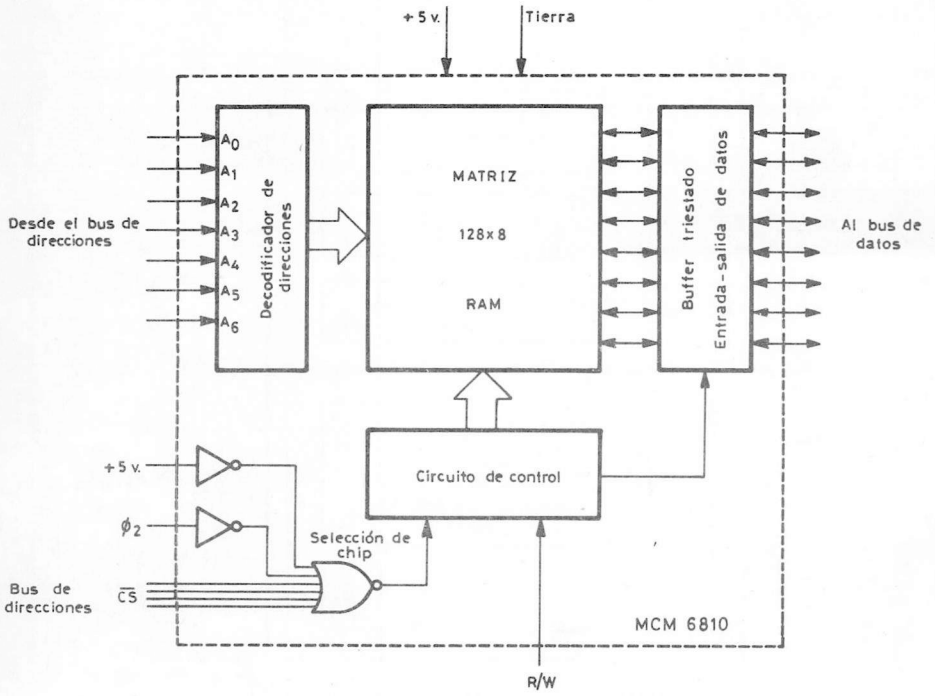


Fig. 7-4.–Estructura de la memoria RAM MCM 6810.

MODULO DE ENTRADA–SALIDA DE DATOS EN PARALELO MC 6820. PIA

La idea de construir un módulo de carácter general para controlar la entrada y salida de datos a la CPU fue propuesta por Motorola con el PIA y, posteriormente, casi todas las familias para microcomputador adoptaron uno de características semejantes.

El PIA está formado por dos puertas (Puerta A y Puerta B) de 8 bits, para entrada y salida de datos, que son tratadas como si fuesen dos posiciones de memoria. Cada puerta se programa como una combinación de entradas y salidas, y se controla independientemente. Se puede asignar como entrada o como salida a cada uno de los 8 bits de cada puerta, siempre que el bit correspondiente de otro registro, denominado de dirección de datos (RDD), contenga un bit 1 o 0. Así, si el RDDA contiene un nivel 1 en su bit de menos peso, el bit de menos peso de la puerta A actuará como salida, y si tuviese cargado un 0, como entrada. Por último, cada puerta, además de disponer de un registro de direccionamiento, tiene otro para control (CR). La constitución básica del PIA se muestra en la figura 7-5.

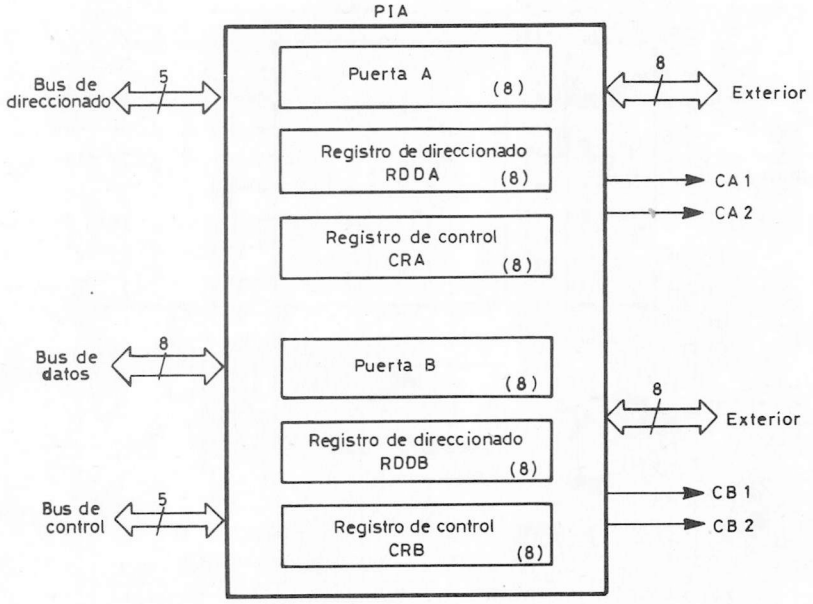


Fig. 7-5.–Estructura simplificada del PIA.

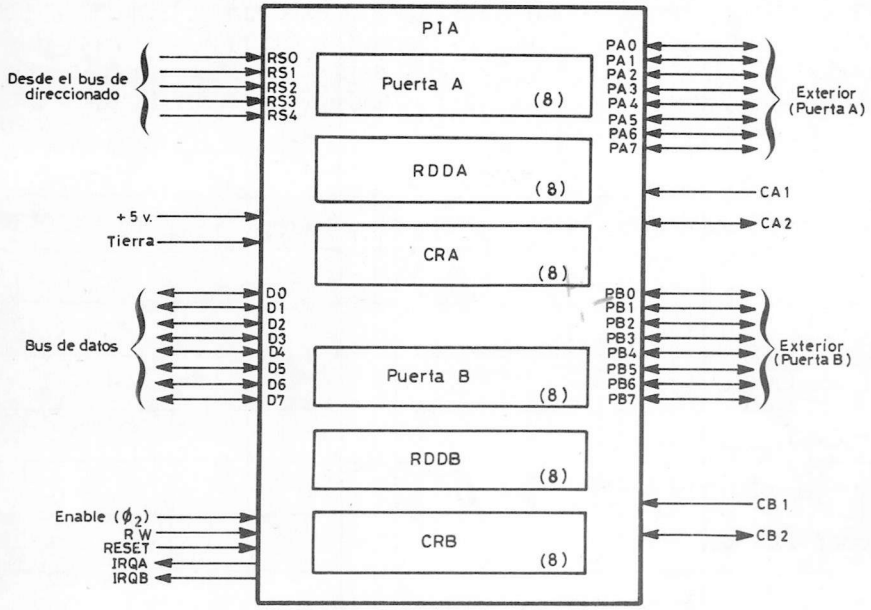


Fig. 7-6.–Diagrama de conexionado del PIA.

Cada una de las líneas del bus de datos está en correspondencia con cada bit de los dos registros, que conforman cada puerta de entrada y salida de datos en paralelo.

El registro de control CR sirve para determinar a las líneas CA_2 y CB_2 de cada puerta como entrada o salida, pudiendo ser programadas, bien para actuar como línea de interrupción, o bien como salida de control del periférico A o B. También CR controla las líneas CA_1 y CB_1 asociadas a cada puerta, como generadoras de interrupciones provenientes de los periféricos controlados.

Una arquitectura más detallada y el diagrama de pines del PIA, se muestran en la figura 7-6.

Las líneas que conectan al PIA con el exterior son 16, que constituyen las puertas A y B para entrada y salida de datos, 2 más CA_1 y CB_1 que pemiten entradas de señales de interrupción desde los periféricos A y B respectivamente y, finalmente, CA_2 y CB_2 sirven para controlar al periférico y se pueden comportar como líneas de entrada de interrupciones o como líneas específicas de salida para control del periférico.

El bus de datos intercambia con la CPU los datos que han de salir o entrar por las puertas.

Para direccionar o seleccionar un chip PIA existen tres líneas CS0, CS1 y CS2. Hay dos líneas más, que reciben información por el bus de direccionamiento, la RS0 y la RS1, y sirven para seleccionar uno de los registros internos, junto con el bit 2 de los registros de control CRA y CRB, según la siguiente tabla.

Registro seleccionado	RSO	RS1	Bit 2 de CRA	bit 2 de CRB
Puerta A	0	0	1	–
Dirección A DDRA	0	0	0	–
Control A CRA	0	1	–	–
Puerta B	1	0	–	1
Dirección B DDRB	1	0	–	0
Control B CRB	1	1	–	–

Así como el bit 2 de los registros de control sirve para la selección de registros internos, el resto de los bits tienen las siguientes misiones, expresadas en la figura 7-7.

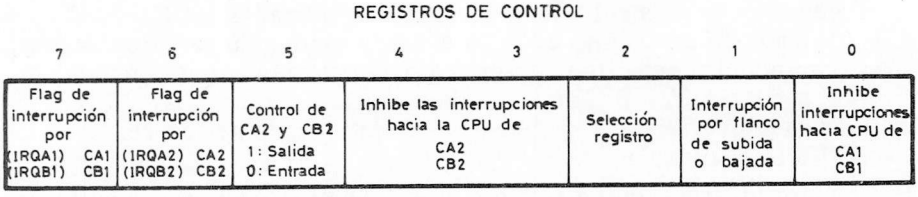


Fig. 7-7.–Misión de los diferentes bits de los registros de control de las puertas A o B.

Las 5 líneas que se conectan al bus de control, IRQA e IRQB, transmiten peticiones de interrupción a la CPU desde los periféricos A o B respectivamente. La señal de habilitación Enable se conecta normalmente a laseñal de reloj ϕ_2, que permite la salida de datos desde la CPU, si la línea R/W tiene nivel 0, o bien, la entrada de datos, si R/W tiene nivel 1. Por último, la línea de RESET pone a cero todos los registros del PIA.

Los 6 registros del PIA son tratados por la CPU como 6 posiciones de memoria, algunas de ellas coincidentes.

Como resumen, se indica la misión de cada uno de los 3 registros diferentes, que constituyen cada una de las puertas de I/O del PIA:

1.°) Los registros de las puertas A y B (ORA y ORB) contienen o reciben los datos de transferencia.

2.°) Los registros DDRA y DDRB indican el sentido de la transferencia (entrada o salida).

3.°) Los registros de control CRA y CRB regulan la actuación de las líneas de control del periférico, monitorizan las interrupciones provenientes de las líneas de control del periférico y direccionan los registros de datos y su transferencia.

MODULO DE ENTRADA–SALIDA DE DATOS EN SERIE MC 6850. ACIA

Se trata de un módulo de I/O que transfiere al exterior los datos en serie, es decir, los datos, que el sistema microprocesador los trata en paralelo, salen y entran a través del ACIA en serie.

Consta de 4 registros:

1.°) Registro de recepción de datos, que llegan en serie a través de una línea conectada con el exterior y los coloca en paralelo sobre el bus de datos.

2.°) Registro de transmisión de datos, que recibe la información en paralelo desde el bus de datos y los saca en serie.

3.°) Registro de Control, que realiza las siguientes misiones:

- Señales de interrupción.
- Longitud de la palabra (7 u 9 bits).
- Determina la velocidad de recepción y transmisión.
- Control de paridad.
- Paro.

4.°) Registro de Estado, que informa sobre las características de la transferencia, tales como errores en formato, en paridad, ausencia de portadora, etc.

En la figura 7-8 se presenta el diagrama de conexionado del ACIA.

Las entradas CS0, CS1 y $\overline{CS2}$ sirven para la selección del chip, mientras que la 4.ª, que se recibe por el bus de direcciones, decodifica uno de

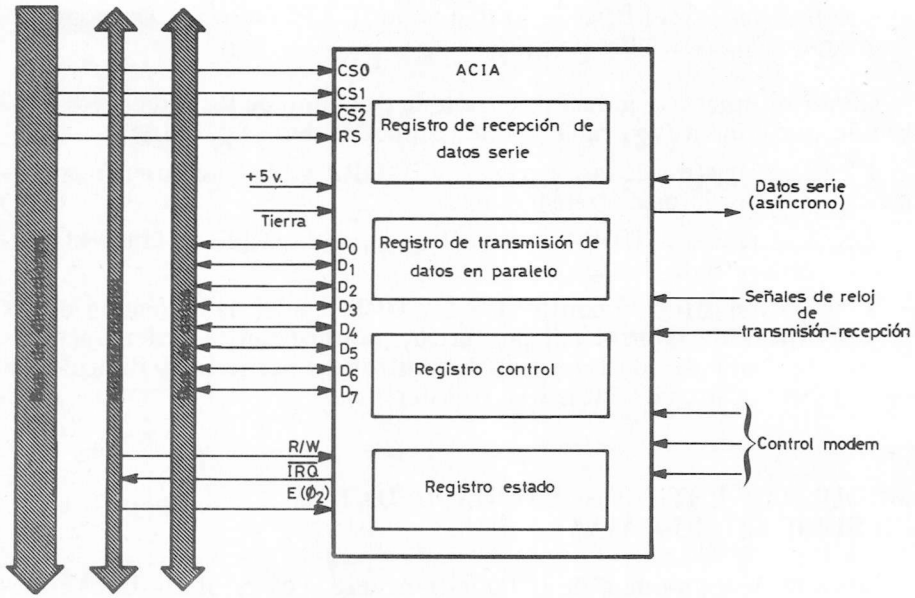


Fig. 7-8.–Diagrama de conexionado del ACIA.

los cuatro registros, puesto que, como sólo se puede escribir en dos y leer en otros dos, con un sólo bit junto con R/W es suficiente.

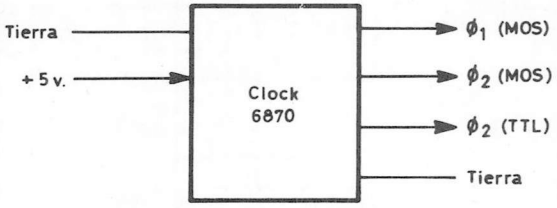
Los datos que entran y salen de la CPU lo hacen a través de las 8 líneas del bus de datos, mientras que los datos que entran o salen al exterior, como lo hacen en serie, sólo precisan de una línea de entrada y otra de salida.

En el bus de control, aparte de R/W está la línea $\overline{IRQ}$, que envía a la CPU la petición de interrupción y la línea de habilitación E, conectada directamente a ϕ_2.

GENERADOR DE LA SEÑAL DE RELOJ MC 6870 y MC 6875

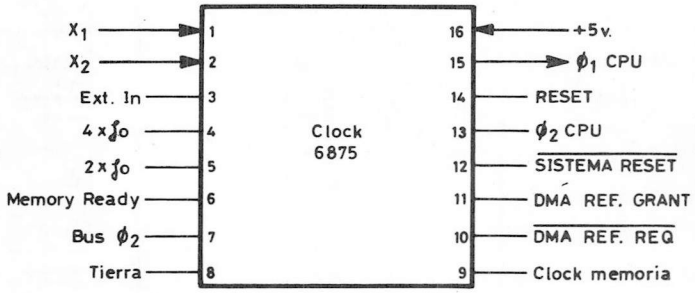
El MC 6870 es un circuito integrado encargado de proporcionar al sistema microcomputador las señales de reloj básicas. Genera dos señales: ϕ_1 y ϕ_2, a nivel MOS, para ser aplicadas directamente al microprocesador, mientras que otra señal ϕ_2, a nivel TTL, puede servir para los demás componentes del sistema que la requieran. Simplificadamente se muestra en la figura 7-9 el diagrama de conexionado del 6870.

Fig. 7-9.–Diagrama del conexionado del clock 6870.



Existe una versión más perfecta, denominada MC 6875, cuyo diagrama se presenta en la figura 7-10.

Fig. 7-10.–Clock 6875.



SISTEMA MINIMO CON LA FAMILIA MICROCOMPUTADOR 6800

Conexionando entre sí a los diversos elementos de la familia 6800, a través de los buses de direccionamiento, de datos y de control, se obtiene un sistema operativo alrededor del microprocesador 6800.

La figura 7-11 muestra un sistema básico formado por la CPU 6800, una ROM 6830, una RAM 6810, un PIA, un ACIA y un clock, tal como lo propone en su información el fabricante.

Un detalle importante en la organización del sistema lo constituye la distribución de las 65 K posiciones de memoria, de capacidad máxima, o sea, el mapa de memoria.

Las 128 primeras posiciones de la memoria, comprendidas entre la 0000 y la 007F, se asignan al chip de RAM, refiriéndonos al sistema de la figura 7-11. De esta forma es posible utilizar el direccionamiento restringido que para direccionar una posición de memoria sólo precisa de 1 byte, detrás del que sigue al del código OP. Esto significa que el byte de más peso que determina la página vale siempre 00. Exclusivamente de esta forma se puede acceder a las 256 primeras posiciones de memoria, que

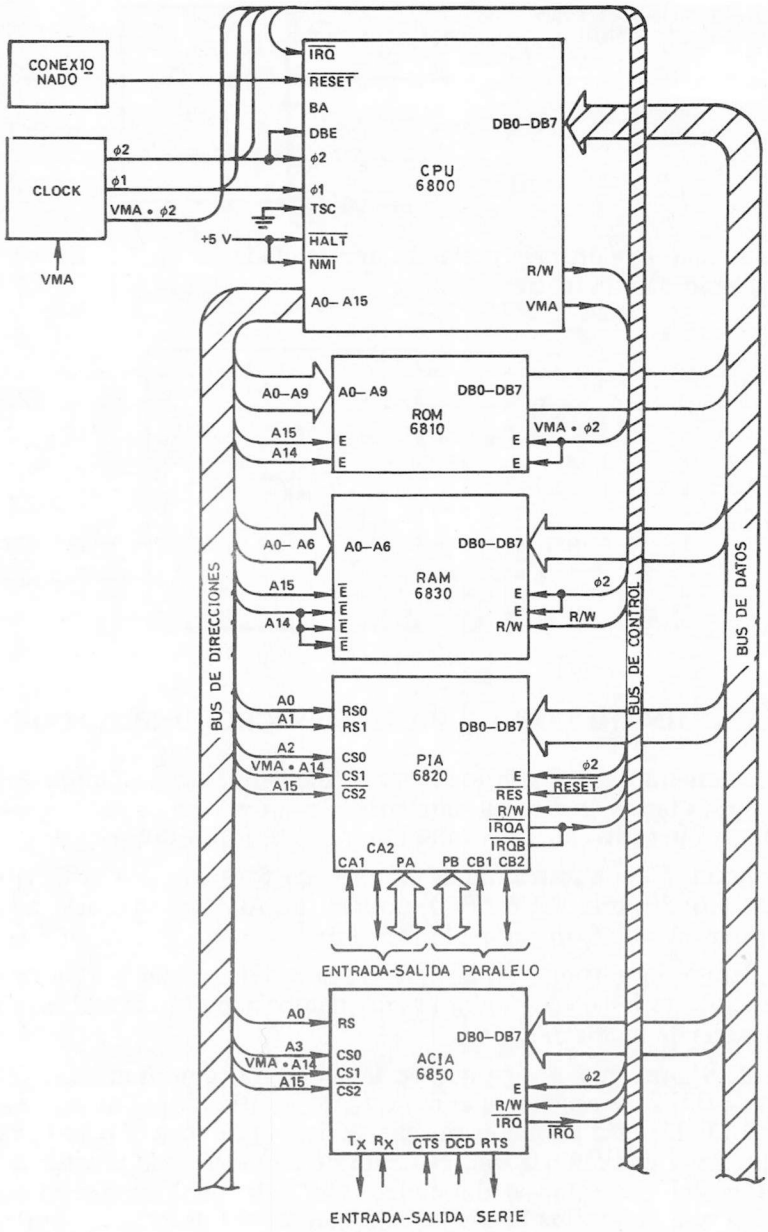


Fig. 7-11.–Sistema mínimo a base del microprocesador M 6800.

constituyen la página cero, que, en el sistema de la figura 7-11, quedan reducidas a 128 al disponer de 1 solo chip de RAM. El direccionado del chip de RAM se efectúa por los 7 bits de menos peso del bus de direcciones ($A_0 - A_6$) y por el A_{14} y el A_{15}, que deben valer cero en este caso.

Para la selección de las posiciones de la ROM de 1 K se emplean los 10 bits de menos peso del bus de direcciones y los bits 14 y 15 del mismo, lo que supone una ocupación del mapa de memoria comprendida entre las posiciones C000 y C3FF.

Las posiciones de memoria reservadas al PIA son 4, desde la 4004 hasta la 4007, lo que exige que el bit 15 del bus de direcciones valga cero, el bit 14 = 1, el A2 = 1 y el A1 y A0 variables, según el registro del PIA que se selecciona con RS0, RS1 y el bit 2 del registro CR, de acuerdo con la siguiente tabla.

DIRECCION	RS1	RS0	CR bit2	Registro del PIA
4004	0	0	0 (A)	DDRA
4004	0	0	1 (A)	Puerta A
4005	0	1	–	CRA
4006	1	0	0 (B)	DDRB
4006	1	0	1 (B)	Puerta B
4007	1	1	–	CRB

El ACIA es gobernado mediante dos posiciones de memoria de la forma siguiente:

DIRECCION	RS	R/W	Registro del ACIA
4008	0	0	Registro de Control
4008	0	1	Registro de Estado
4009	1	0	Registro de transmisión
4009	1	1	Registro de recepción

La figura 7-11 resulta simplificada al no estar representada la tensión de alimentación de +5 V, necesaria en todos los módulos. El clock proporciona una señal que, mediante un operador AND, se transforma en VMA.ϕ_2 utilizado para determinar los ciclos máquina en los que se realiza intercambio de información con las posiciones de memoria, considerando como tales los registros del PIA y del ACIA.

Motorola ofrece tarjetas de circuito impreso, tales como la M68MM01, cuya foto aparece en la figura 7-12, en la que viene implementado un sistema de microprocesador básico, formado por la CPU 6800 (U_{16}), 4 K bytes de ROM (U_{12}, U_{13}, U_{14} y U_{15}), 1 K de RAM estática, 3 PIAS (U_{20}, U_{21} y U_{22}), que proporcionan 6 puertas de 8 bits para entrada y salida de datos y un generador de señales de reloj.

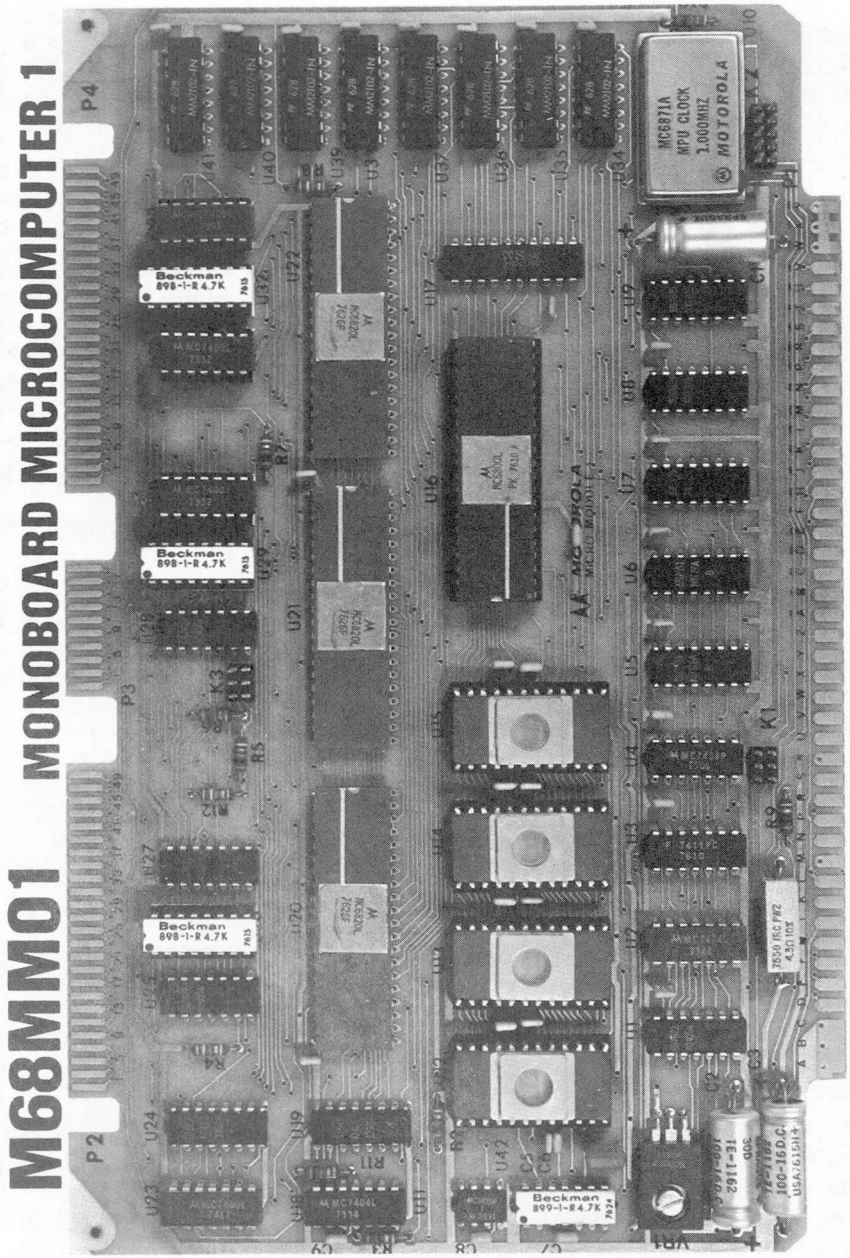


Fig. 7-12–Foto de una tarjeta de circuito impreso que contiene un sistema básico con microprocesador.

MOTOROLA Semiconductors
BOX 20912 • PHOENIX, ARIZONA 85036

MCM6810A
(0 to 70°C; L or P Suffix)

MCM6810AC
(−40 to 85°C; L Suffix only)

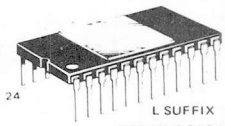
128 X 8-BIT STATIC RANDOM ACCESS MEMORY

The MCM6810 is a byte-organized memory designed for use in bus-organized systems. It is fabricated with N-channel silicon-gate technology. For ease of use, the device operates from a single power supply, has compatibility with TTL and DTL, and needs no clocks or refreshing because of static operation.

The memory is compatible with the M6800 Microcomputer Family, providing random storage in byte increments. Memory expansion is provided through multiple Chip Select inputs.

- Organized as 128 Bytes of 8 Bits
- Static Operation
- Bi-Directional Three-State Data Input/Output
- Six Chip Select Inputs (Four Active Low, Two Active High)
- Single 5-Volt Power Supply
- TTL Compatible
- Maximum Access Time = 350 ns – MCM6810AL1
  450 ns – MCM6810AL

MOS
(N-CHANNEL, SILICON-GATE)
128 X 8-BIT STATIC
RANDOM ACCESS MEMORY

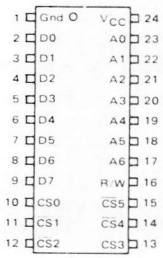


L SUFFIX
CERAMIC PACKAGE
CASE 716

NOT SHOWN

P SUFFIX
PLASTIC PACKAGE
CASE 709

PIN ASSIGNMENT

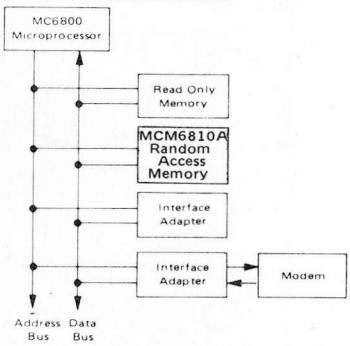


ABSOLUTE MAXIMUM RATINGS (See Note 1)

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 to +7.0	Vdc
Input Voltage	V_{in}	-0.3 to +7.0	Vdc
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature Range	T_{stg}	-65 to +150	°C

NOTE 1 Permanent device damage may occur if ABSOLUTE MAXIMUM RATINGS are exceeded. Functional operation should be restricted to RECOMMENDED OPERATING CONDITIONS. Exposure to higher than recommended voltages for extended periods of time could affect device reliability.

M6800 MICROCOMPUTER FAMILY
BLOCK DIAGRAM



MCM6810A RANDOM ACCESS MEMORY
BLOCK DIAGRAM

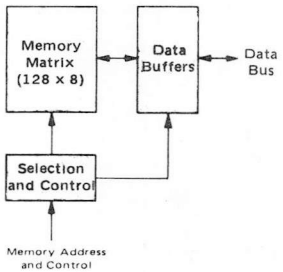


Fig. 7-13.–Características de la memoria MCM 6810, ofrecidas por su fabricante MOTOROLA.

MOTOROLA Semiconductors
BOX 20912 . PHOENIX, ARIZONA 85036

MCM6830A

Advance Information

MOS
(N-CHANNEL, SILICON-GATE)

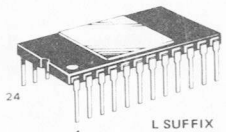
1024 X 8-BIT
READ ONLY MEMORY

1024 X 8-BIT READ ONLY MEMORY

The MCM6830A is a mask-programmable byte-organized memory designed for use in bus-organized systems. It is fabricated with N-channel silicon-gate technology. For ease of use, the device operates from a single power supply, has compatibility with TTL and DTL, and needs no clocks or refreshing because of static operation.

The memory is compatible with the M6800 Microcomputer Family, providing read only storage in byte increments. Memory expansion is provided through multiple Chip Select inputs. The active level of the Chip Select inputs and the memory content are defined by the customer.

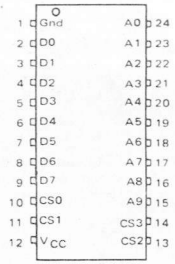
- Organized as 1024 Bytes of 8 Bits
- Static Operation
- Three-State Data Output
- Four Chip Select Inputs (Programmable)
- Single 5-Volt Power Supply
- TTL Compatible
- Maximum Access Time = 500 ns



L SUFFIX
CERAMIC PACKAGE
CASE 716

NOT SHOWN: P SUFFIX
PLASTIC PACKAGE
CASE 709

PIN ASSIGNMENT

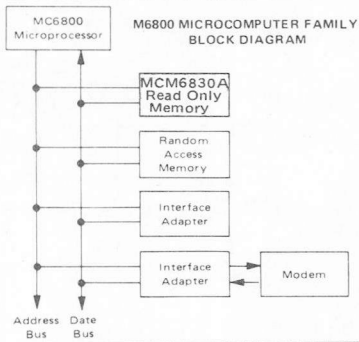


ABSOLUTE MAXIMUM RATINGS (See Note 1)

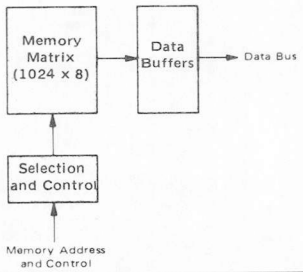
Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 to +7.0	Vdc
Input Voltage	V_{in}	-0.3 to +7.0	Vdc
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

NOTE 1. Permanent device damage may occur if ABSOLUTE MAXIMUM RATINGS are exceeded. Functional operation should be restricted to RECOMMENDED OPERATING CONDITIONS. Exposure to higher than recommended voltages for extended periods of time could affect device reliability.

M6800 MICROCOMPUTER FAMILY
BLOCK DIAGRAM



MCM6830A READ ONLY MEMORY
BLOCK DIAGRAM



This is advance information and specifications are subject to change without notice.

Fig. 7-14.– Características de la memoria MCM 6830, ofrecidas por MOTOROLA.

MOTOROLA Semiconductors
BOX 20912 • PHOENIX, ARIZONA 85036

MCM6832

Advance Information

MOS

(N-CHANNEL, LOW THRESHOLD)

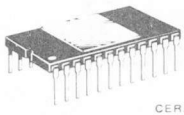
2048 x 8-BIT
READ ONLY MEMORY

2048 x 8-BIT READ ONLY MEMORY

The MCM6832 is a mask programmable byte-organized memory designed for use in bus-organized systems. It is fabricated with N-channel metal-gate technology. For ease of use, the device is compatible with TTL and DTL, and needs no clocks or refreshing because of static operation.

The memory is compatible with the M6800 Microcomputer Family, providing read only storage in byte increments. Memory expansion is provided through a Chip Select input. The active level of the Chip Select input and the memory content are defined by the customer.

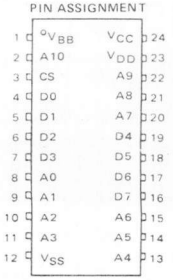
- Organized as 2048 Bytes of 8 Bits
- Static Operation
- Three-State Data Output
- Programmable Chip Select
- TTL Compatible
- Maximum Access Time = 500 ns



L SUFFIX
CERAMIC PACKAGE
CASE 716

NOT SHOWN: P SUFFIX
PLASTIC PACKAGE
CASE 709

PIN ASSIGNMENT

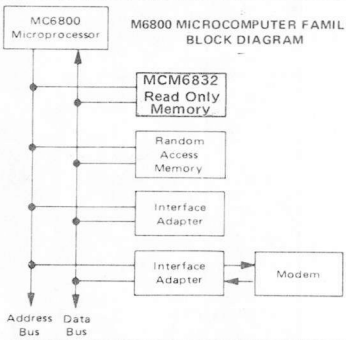


ABSOLUTE MAXIMUM RATINGS[1] (Referenced to V_{SS})

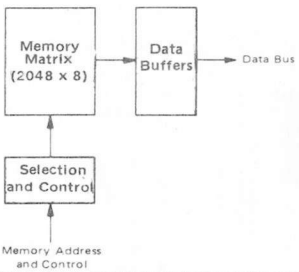
Rating	Symbol	Value	Unit
Supply Voltages	V_{DD} V_{CC} V_{BB}	-0.3 to +15 -0.3 to +6.0 -10 to +0.3	Vdc
Address/Control Input Voltage	V_{in}	-0.3 to +15	Vdc
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature Range	T_{stg}	-55 to +125	°C

Note 1. Permanent device damage may occur if ABSOLUTE MAXIMUM RATINGS are exceeded. Functional operation should be restricted to RECOMMENDED OPERATING CONDITIONS. Exposure to higher than recommended voltages for extended periods of time could affect device reliability.

M6800 MICROCOMPUTER FAMILY
BLOCK DIAGRAM



MCM6832 READ ONLY MEMORY
BLOCK DIAGRAM

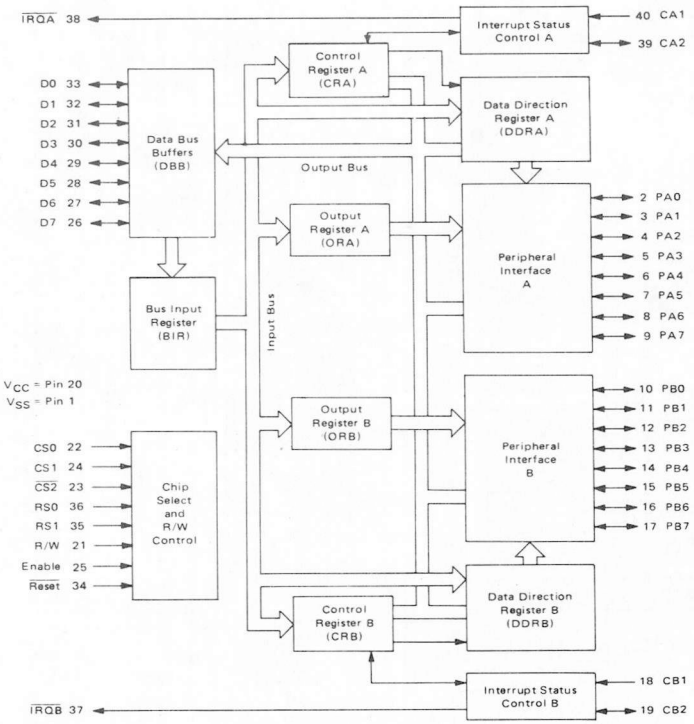


This is advance information and specifications are subject to change without notice.

Fig. 7-15.–Características de la memoria MCM 6832, ofrecidas por Motorola.

MC6820

EXPANDED BLOCK DIAGRAM



MOTOROLA Semiconductor Products Inc.

Fig. 7-16.–Arquitectura del PIA 6820, ofrecidas por MOTOROLA.

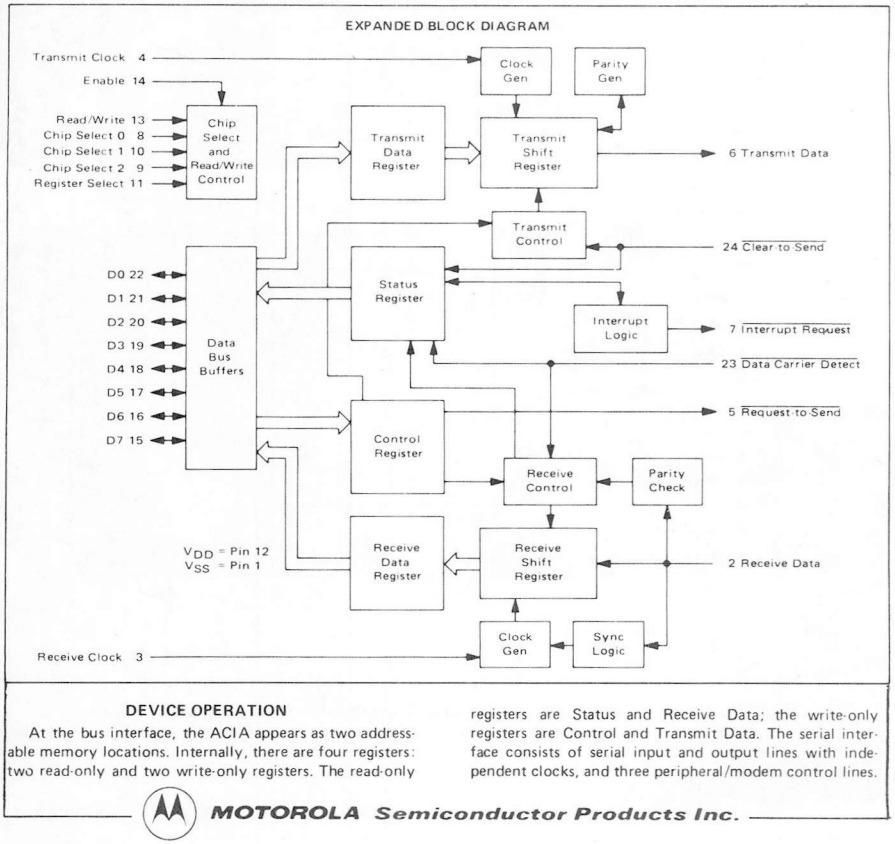


Fig. 7-17.–Diagrama por bloques del ACIA 6850, ofrecido por su fabricante.

MODERNO SISTEMA MINIMO CON 2 CHIPS

Motorola ofrece en la actualidad un versátil microcomputador con 2 chips, compatible con todos los periféricos de la familia M 6800. Estos 2 potentes chips son:

MC6802: Que incluye además de la CPU, circuito interno de reloj y 128 bytes de RAM.

MC6846: Dispone de 2 K bytes de ROM, un contador/temporizador programable de 16 bits y 10 líneas de entrada y salida.

La figura 7-17 bis muestra la interconexión de los 2 chips para implementar la configuración del microcomputador.

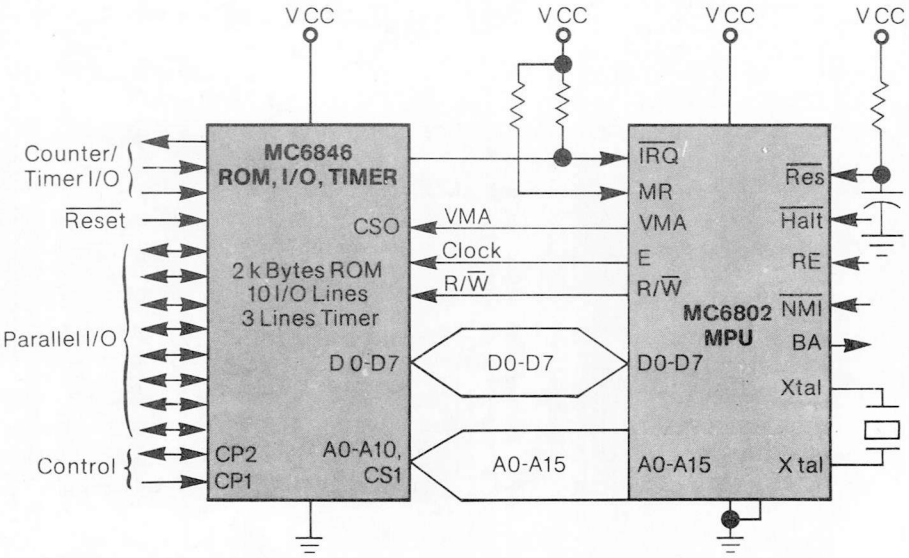


Fig. 7-17 bis.–Configuración de un microcomputador mediante sólo 2 chips.

2.ª PARTE: FAMILIA MICROCOMPUTADOR R-6500

ELEMENTOS QUE COMPONEN LA FAMILIA R 6500

De forma similar a la exposición que se ha hecho de la familia Motorola, se presentan a continuación los componentes más representativos, que complementan el trabajo de los microprocesadores R 65XX para configurar sistemas microcomputadores. Se describen los circuitos integrados más característicos en cada uno de los 3 tipos siguientes:

a) Memorias: Tipo RAM, estático, modelo R 2114 de 4 K bits, organizados en 1.024 bytes de 4 bits cada uno. Tipo ROM estática, modelo R 2332 de 32 K bits, distribuidos en 4 K bytes de 8 bits cada uno. Memorias de burbujas.

b) Módulos de Entrada y salida: El PIA R 6520 y el VIA R 6522.

c) Modulos mixtos: Rockwell ha implementado sobre el mismo chip todos los bloques del sistema microcomputador, que precisan sus microprocesadores R 65 XX, de esta forma, un sistema básico y ampliable puede componerse tan sólo de 2 chips. Destacamos el modelo R 6530, que contiene RAM, ROM I/O y TIMER, el cual debe ser grabado en fábrica por la presencia de la memoria ROM. Otro modelo parecido, pero sin memoria ROM, es el R 6532.

MEMORIA RAM R 2114

Este circuito integrado de 18 pines contiene una matriz de memoria de 1.024 palabras de 4 bits, con un tiempo de acceso máximo de 450 ns y sin necesidad de señales de reloj ni líneas de strobe. Su alimentación es de +5 V, totalmente compatible con la familia TTL. En la figura 7-18 se presenta el diagrama de pines y de conexionado de esta memoria.

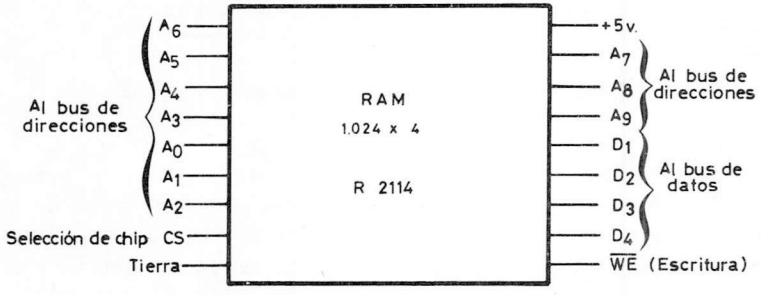


Fig. 7-18.–Diagrama de conexionado de la memoria R 2114.

El R 2114 dispone de 10 líneas de entrada, que direccionan la posición elegida a través del bus de direccionamiento (A_0, A_1, ..., A_9). Existen 4 líneas para efectuar la entrada y salida de datos (D_1, D_2, D_3 y D_4). Finalmente, existen 2 pines para la entrada de la alimentación, uno más para la selección de chip, cuando hay más de uno (CS) y un último, que determina si la operación, que se ha de realizar, es de escritura o lectura.

MEMORIA ROM R 2332

Se trata de una memoria ROM, de canal N, estática (no precisa señal de reloj), preparada para trabajar en la familia microcomputador R 6500 a frecuencias de 1 y 2 MHz. Su matriz consta de 4.096 bytes de 8 bits cada uno y su tiempo de acceso es de 250 ns, requiriendo una sola alimentación de +5 V. En la figura 7-19 se muestra el diagrama de conexionado de este chip.

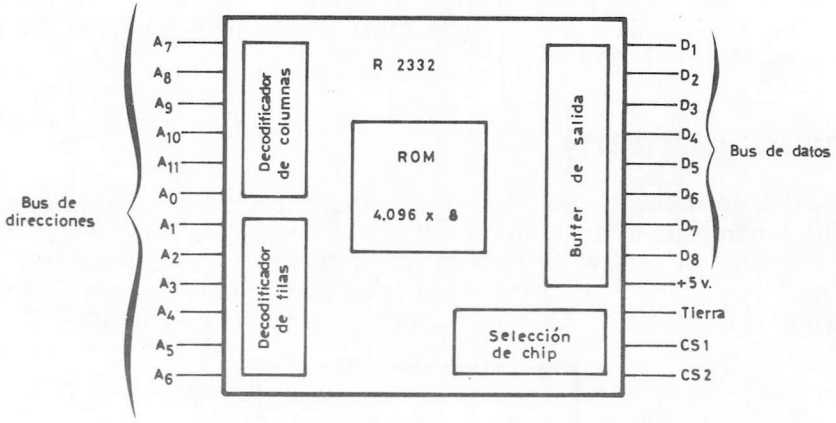


Fig. 7-19.–Diagrama de conexiones de la memoria R 2332.

El R 2332 dispone de 12 líneas conectadas al bus de direcciones, que seleccionan la posición de memoria, 8 líneas más conectadas al bus de datos, que sacan al exterior el contenido fijo de las posiciones de memoria. Por último, dispone de 2 líneas para la entrada de alimentación, y dos más (CS1 y CS2) para seleccionar 1 chip de entre un máximo de 4.

MEMORIAS DE BURBUJAS

Las burbujas magnéticas consisten en pequeñísimas zonas magnetizadas, formadas sobre una película muy fina con substrato de granate y bajo la actuación de un campo magnético, siendo capaces de almacenar una elevadísima cantidad de información por unidad de superficie de manera absolutamente estática.

En la fina película magnética cada átomo de hierro y de otros elementos actúan como pequeños imanes, dotados de sus polos Norte y Sur. Inicialmente, estos átomos imantados se colocan en línea, perpendicularmente a la superficie del plano del material, de forma que la mitad de los mismos quedan apuntando el polo Norte hacia arriba y la otra mitad hacia abajo. Debido a las fuerzas de atracción interatómica, cada átomo se alinea en paralelo con el de al lado, formando los denominados "dominios", en que grupos de imanes atómicos se agrupan conjuntamente todos con el polo Norte hacia arriba, o todos con el polo Norte hacia abajo.

Al actuar un campo magnético externo sobre los dominios formados, aquellos cuya orientación coincide con dicho campo se expanden, mientras que los orientados en sentido opuesto se reducen, tanto más cuanto más intenso es el campo magnético, formando las áreas circulares, que reciben el nombre de "burbujas". La presencia de la burbuja representa un nivel digital 1 y la ausencia de la misma un nivel 0. Las burbujas desaparecen cuando el campo supera cierto límite. Cuando el campo varía de intensidad con el plano del cristal, se mueven las burbujas según el gradiente del campo.

Actualmente, el diámetro de cada burbuja es de unas 3 micras, pero es previsible una notable reducción del mismo en pocos años.

El objetivo que trata de alcanzar esta nueva tecnología es la de sustituir los costosos y voluminosos sistemas de almacenamiento de discos usados en la actualidad.

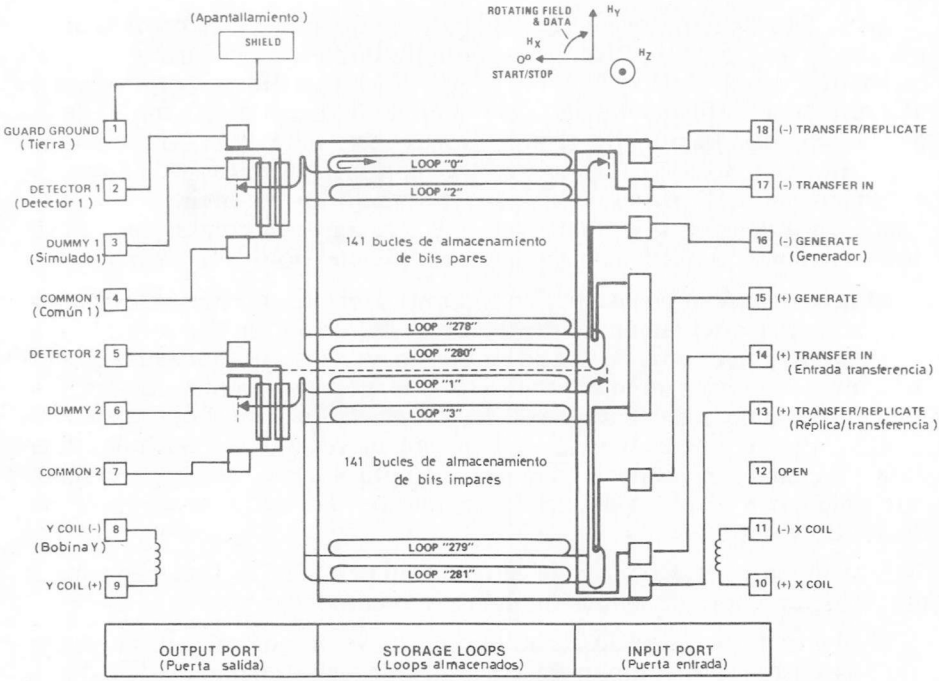
Rockwell recibe una importante ayuda en su proyecto de desarrollo e investigación de las memorias de burbujas por parte de la NASA y las Fuerzas Aéreas de USA, dada la importancia del mismo.

MEMORIA DE BURBUJA RBM 256

La memoria de burbuja RBM 256, fabricada por Rockwell, almacena 266.500 bits y representa uno de los mayores avances en cuanto a materiales, arquitectura y encapsulado.

La RBM 256 se compone de 282 bucles, cada uno de los cuales contiene 1.025 posiciones de burbuja. El elemento opera con un bloque de datos de 260 bits. Transfiere los datos a 150 KHz, tomando menos de 4 ms en acceder al primer bit de un bloque.

Con encapsulado plástico de 18 pines, ocupa una superficie cuadrada menor de 3 cm de lado, con un consumo de 820 mw y un margen de



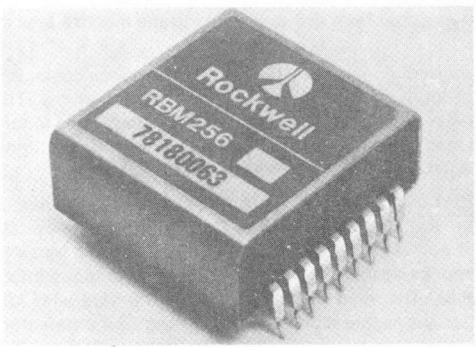


Fig. 7-19 bis.– Arquitectura y foto de la memoria de burbujas RBM 256.

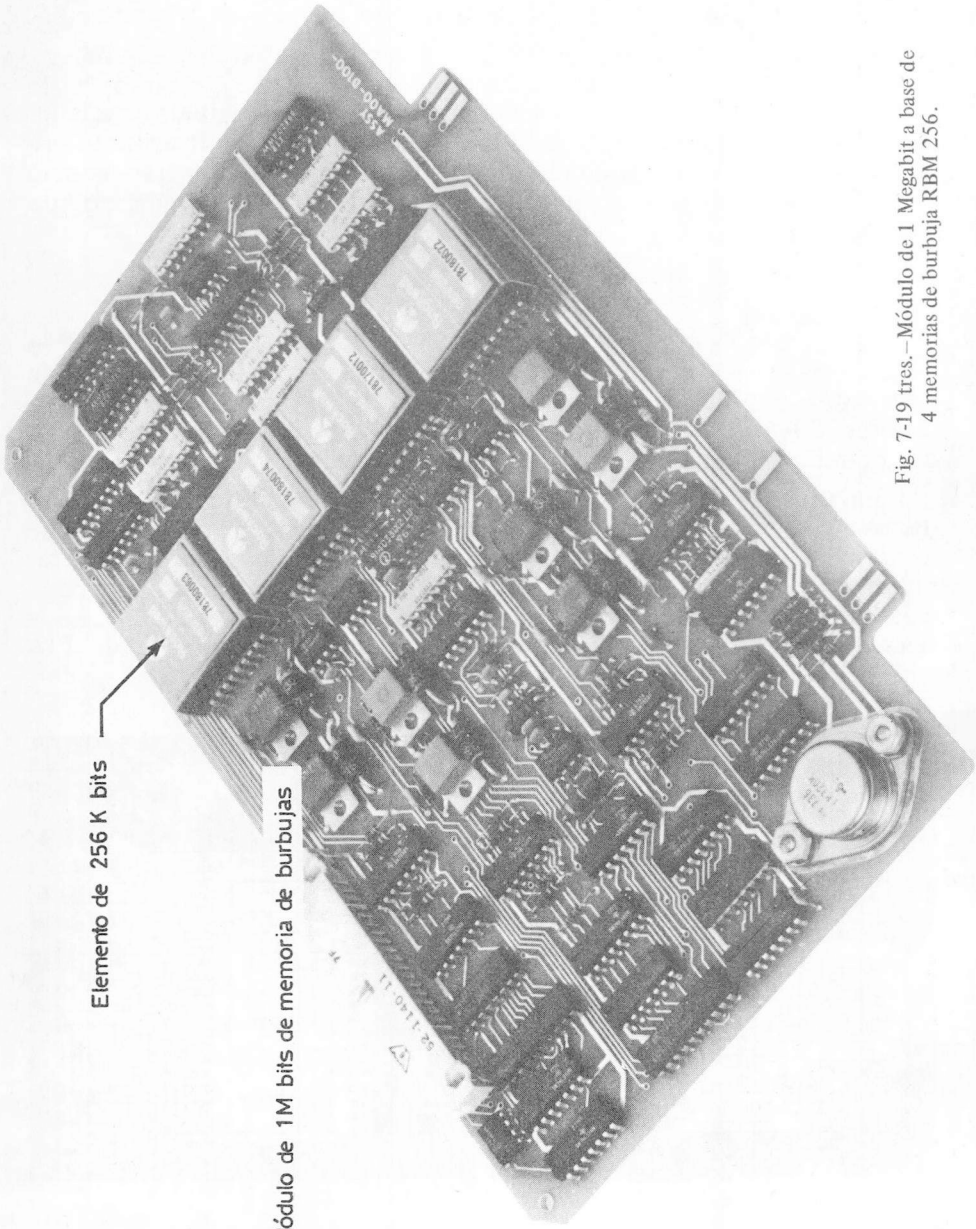


Fig. 7-19 tres.–Módulo de 1 Megabit a base de 4 memorias de burbuja RBM 256.

temperatura entre −10°C hasta 70°C. Su precio actual (1979) de 500 $ para pequeñas cantidades es probable que se reduzca en breve tiempo.

Combinando 4 memorias RBM 256 se fabrica el módulo RLM 658 de 1 Megabit de capacidad, que se ha diseñado para poder alcanzar hasta 2 Megabytes con 16 módulos. Opera a 100 Kbytes por segundo y es compatible con el Sistema 65, basado en el microprocesador R 6502 y bastantes sistemas microcomputadores de desarrollo, que operan con el 6800. En la actualidad, su precio asciende a 2.500 $ por cada módulo RLM 658.

PIA R 6520

Se trata de un chip, que realiza la función de un módulo adaptador de las entradas y salidas del sistema microcomputador con los periféricos exteriores. Sus características son similares al PIA 6820, estudiado dentro de la familia Motorola, por lo que se pueden resumir.

Fundamentalmente, consta de 2 puerta de 8 bits cada una, denominadas A y B, que están controladas por dos registros cada una, uno de direccionamiento, que permite trabajar a cada uno de los bits de las puertas como entrada o como salida y otro registro de control con parecidas funciones al 6.820. Existen dos versiones a 1 MHz y a 2 MHz. En la figura 7-20 se muestra el diagrama simplificado del conexionado del PIA y el de los pines que lo componen.

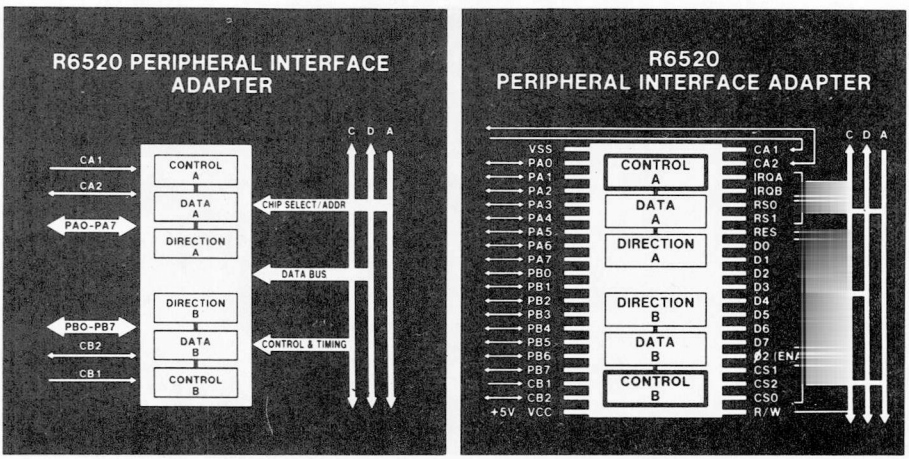


Fig. 7-20.–Diagrama simplificado de conexiones y de pines del PIA.

Existen 4 líneas especiales CA1, CA2, CB1 y CB2, que se utilizan para el control de los periféricos y de sus interrupciones, las cuales se manejan con los registros de control, según se expresa en la figura 7-21 y en la tabla 7-22.

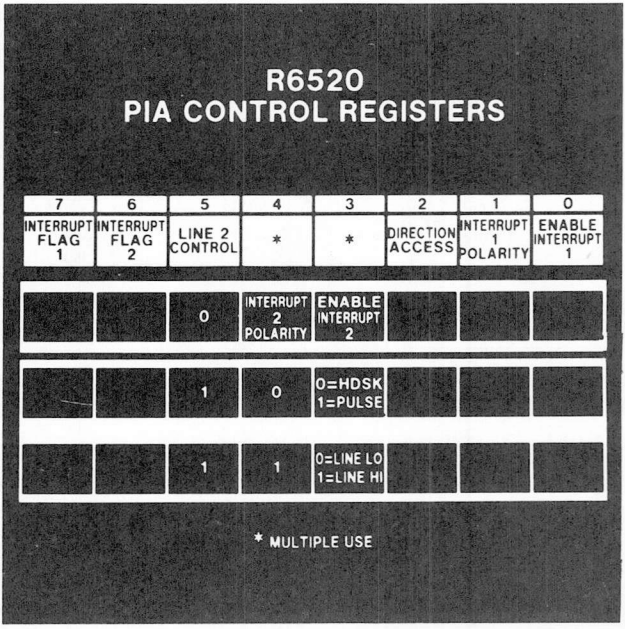


Fig. 7-21.–Funciones de los bits de los registros de control del 6520.

SUMARIO DEL FUNCIONAMIENTO DEL R6520

CA1/CB2 Control

CRA (CRB)		Paso a activa de la Señal de Entrada*	IRQA (IRQB) Salidas de Interrupción
Bit 1	Bit 0		
0	0	Negativa	Inhabilita – Permanece Alta
0	1	Negativa	Habilita – Baja cuando el bit 7 en CRA (CRB) es set, por transición activa de la señal en CA1 (CB1)
1	0	Positiva	Inhabilita – Permanece Alta
1	1	Positiva	Habilita – Según lo explicado arriba

* *Nota:* El bit 7 de la CRA (CRB) es 1, al pasar a activa la señal CA1 (CB1). Esto es independiente del estado del bit 0 en CRA (CRB).

En la tabla 7-22 el fabricante ofrece los diferentes códigos de los bits de los registros de control CRA y CRB para los diversos modos de trabajo de las líneas CA1, CA2, CB1 y CB2.

Modos de Entrada CA2/CB2

CRA (CRB)			Paso a activa de la Señal de Entrada*	IRQA (IRQB) Salidas de Interrupción
Bit 5	Bit 4	Bit 3		
0	0	0	Negativa	Inhabilita – Permanece Alta
0	0	1	Negativa	Habilita – Baja cuando el bit 6 en CRA (CRB) es 1, a causa de la transición de la señal en CA2 (CB2)
0	1	0	Positiva	Inhabilita – Permanece Alta
0	1	1	Positiva	Habilita – Según lo explicado arriba.

* *Nota:* El bit 6 de CRA (CRB) se pone a 1 al cambiar a activa la señal CA2 (CB2). Esto es independiente del estado del bit 3 en CRA (CRB).

Modos de Salida CA2

CRA			Modo	Descripción
Bit 5	Bit 4	Bit 3		
1	0	0	"Handshake" en lectura	CA2 se pone alto en el cambio a activo de la señal de entrada de interrupción CA1 y se pone bajo por medio de una operación "lectura de Datos A" del microprocesador. Esto permite el control de transferencia de datos desde el periférico al microprocesador.
1	0	1	Salida Impulso	CA2 baja un ciclo después de una operación de "Lectura de Datos A". Este impulso puede usarse para indicar al periférico que tomaba el dato.
1	1	0	Salida Manual	CA2 bajo
1	1	1	Salida Manual	CA2 alto

* *Nota:* "Handshake" significa "Acuse de recibo".

Modos de Salida CB2

CRB			Modo	Descripción
Bit 5	Bit 4	Bit 3		
0	0	0	"Handshake" en escritura	CB2 es bajo en la operación de "Escritura de Datos B" y pasa a alto cambiando a activa la señal de entrada de interrupción del CB1. Esto permite un control en la transferencia de datos del microprocesador al periférico.
1	0	1	Salida Impulso	CB2 baja un ciclo después de una operación de "Escritura de Datos B" del microprocesador. Señala al periférico qué es útil al dato.
1	1	0	Salida Manual	CB2 bajo
1	1	1	Salida Manual	CB2 alto

Fig. 7-22.—Tabla de operación de los registro de control del 6520.

VIA. R 6522

Este chip tiene funciones similares al PIA R 6520. Dispone, además, de 2 contadores/temporizadores programables de 16 bits y un registro, que permite la adaptación de información paralelo/paralelo-serie entre los elementos más destacados. En la figura 7-23 se muestra un diagrama

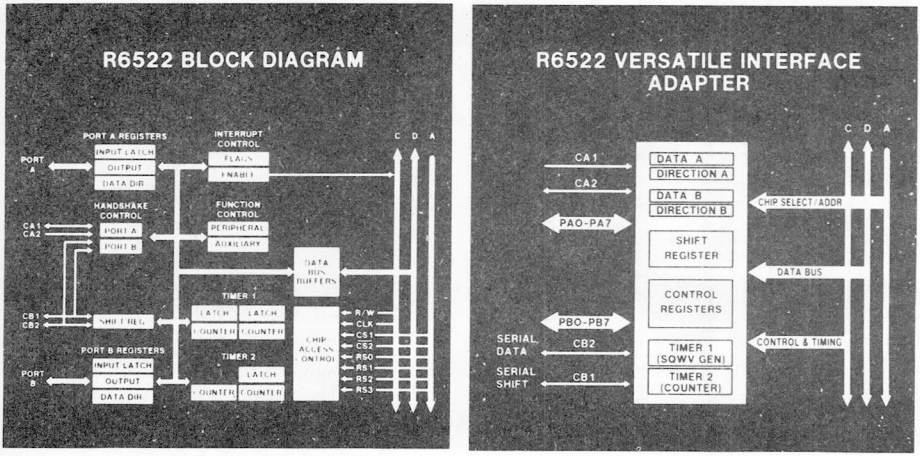


Fig. 7-23.—Diagrama por bloques del R 6522 y adaptación del mismo con los buses del sistema y con los periféricos.

por bloques del R 6522 y otro con la adaptación a los buses del sistema y con los periféricos exteriores.

La inclusión de 2 TIMER (contadores-temporizadores), registro de desplazamiento, posibilidad de entrada de datos en serie y, en general, un control y una adaptación a los periféricos muy potente hacen muy útil al VIA en muchas aplicaciones.

R 6530 (ROM/RAM/I-O/TIMER) Y R 6532 (RAM/I-O/TIMER)

El R 6530 está diseñado para formar un sistema básico microprocesado con una CPU de la familia R 6500. Contiene una memoria ROM de 1 K X 8 de capacidad, que debe ser grabada en fábrica, por lo que el chip sólo resultará interesante cuando se utilice en grandes series. También dispone de una RAM de 64 x 8 bits, 2 puertas bidireccionales de 8 bits, programables para entrada y salida de datos y, finalmente, de un TIMER programable por software, capaz de controlar tiempos comprendido entre 1 y 262.144 períodos de reloj. El R 6532 es similar al R 6530, pero sin memoria ROM, por lo que se puede emplear en la construcción de prototipos o series cortas en combinación con una memoria ROM.

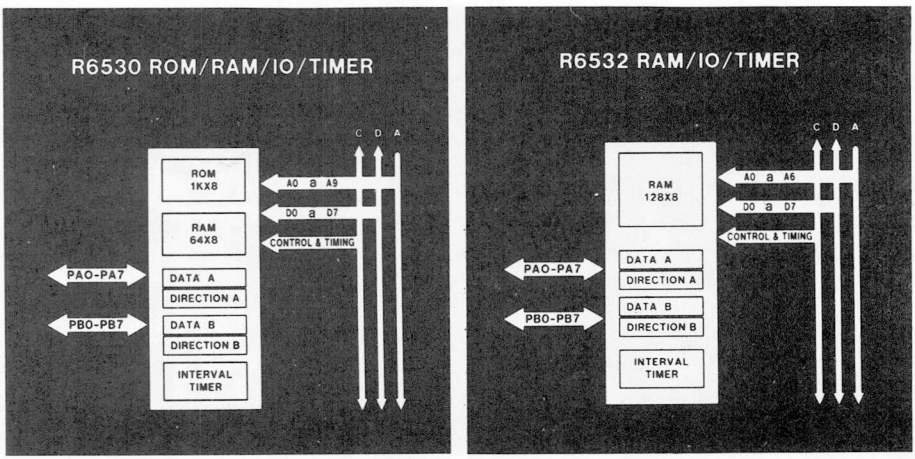


Fig. 7-24.–Diagrama por bloques e interface de los chips R 6530 y R 6532.

En la figura 7-25 se presenta el diagrama de pines del R 6530 y en la 7-26 una arquitectura más detallada de los bloques que lo forman.

Dentro de la arquitectura del 6530, mostrada en la figura 7-26, debe tenerse en cuenta que CS1 y CS2 son opciones mascarables en lugar de PB5, y PB6.PB7 puede usarse como línea $\overline{IRQ}$.

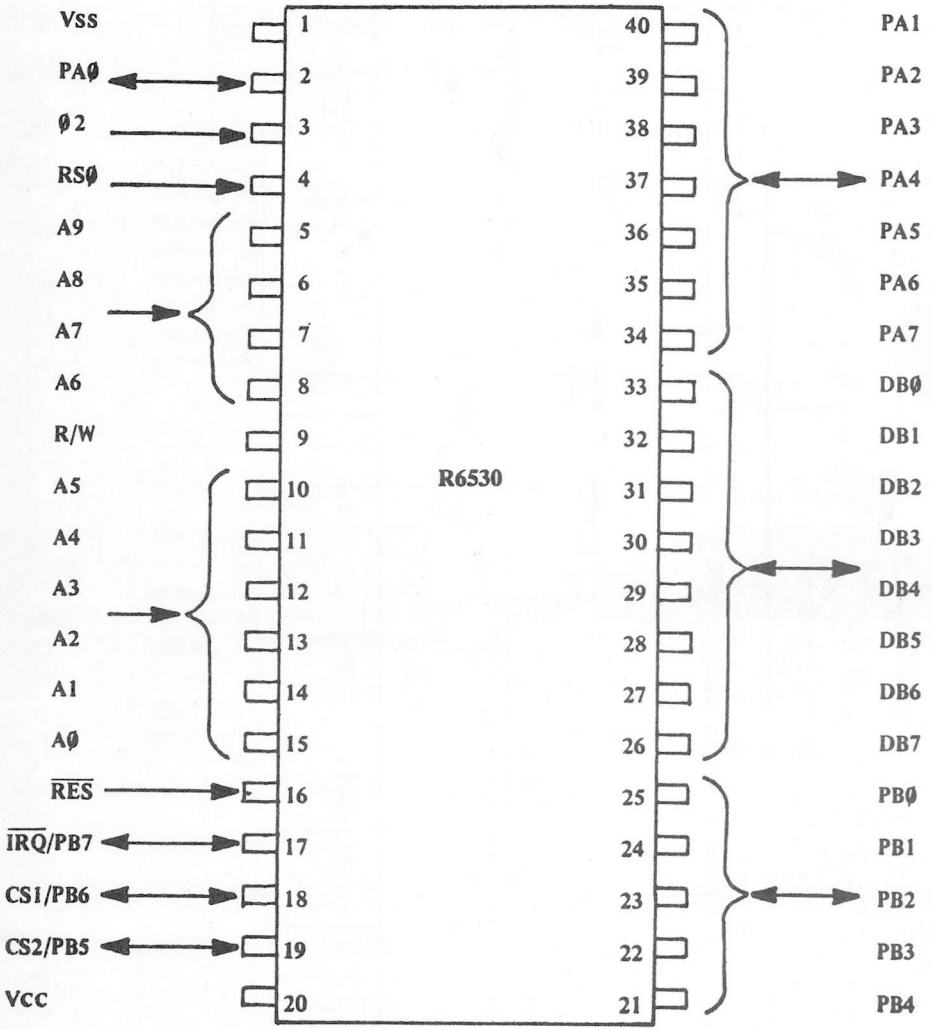


Fig. 7-25.–Designación del conjunto de pines del 6530.

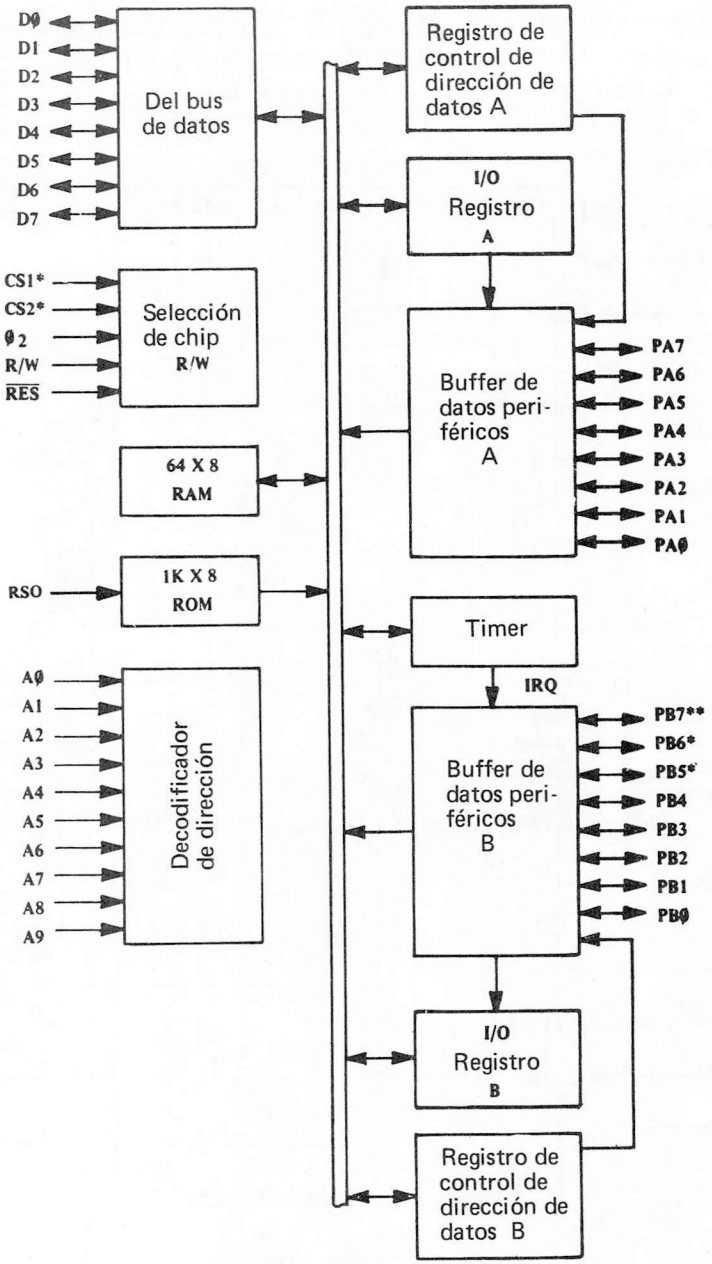


Fig. 7-26.–Arquitectura interna del R 6530.

La actuación de los pines del R 6530, de forma resumida, es la siguiente:

$\overline{RES}$: Inicializa los registros de I/O, haciendo que las puertas actúen como entradas. Realiza una puesta a cero.

R/W: Permite la lectura o escritura del microprocesador al 6530.

$\overline{IRQ}$: Produce una interrupción IRQ desde el temporizador. Si no se usa como interrupción, actúa como pin de I-O (PB7).

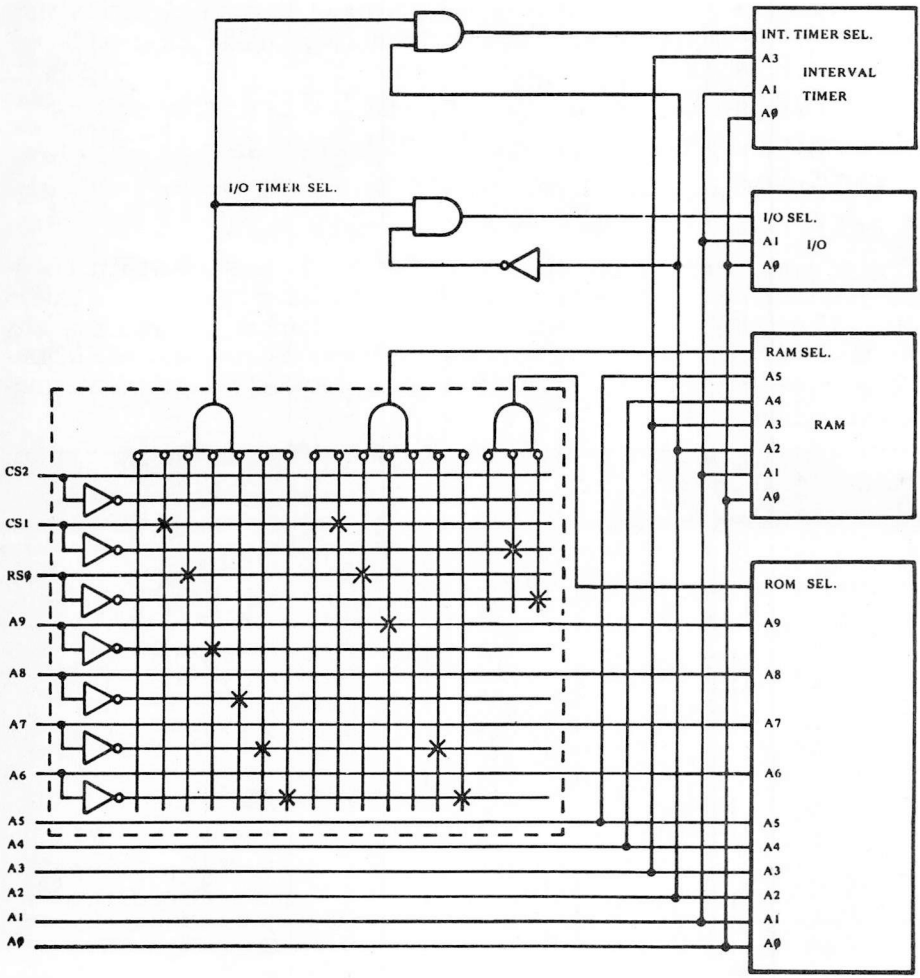


Fig. 7-27.–Direccionado de un chip R 6530.

D_0-D_7: Son los 8 bits del bus de datos bidireccional.

PA0-PA7 y PB0-PB7: Son los 16 pines que conectan al sistema con los periféricos. Pueden actuar como entradas o salidas, según que el bit correspondiente del registro de direccionamiento esté cargado con un 0 o con un 1.PB5, PB6 y PB7 tienen los usos especiales ya indicados.

A0-A9: Son 10 líneas de direccionamiento. Además, en la ROM se usa también el pin RS (ROM SEL.). Para selección de chip, en caso de usar varios 6530, existen las líneas CS1 y CS2 (PB5 y PB6), que actúan independientemente. En la figura 7-27 se muestra el diagrama de conexionado para el direccionamiento de un chip 6530 en el cual la selección de cada bloque ha quedado de la siguiente forma:

La ROM queda seleccionada por CS1.RS0.

La RAM queda seleccionada por $\overline{CS1}.\overline{RS0}.\overline{A9}$.A7.A6.

El I/O, TIMER se selecciona con $\overline{CS1}.\overline{RS0}$. A9.A8.A7.A6.

En este ejemplo CS2 puede ser usado como PB5.

Si se precisa direccionar a 7 chips 6530, los 7 K de ROM deberán ocupar posiciones comprendidas entre 1.025 y 65.536, para lo cual las líneas A13, A14 y A15 deberán estar a nivel 1. En cambio, si direccionamos RAM o I-O, estas 3 líneas del bus de direccionamiento estarán a 0. Deberán emplearse las líneas CS1 y CS2 conectadas a A11 y A12; RSO quedará conectada a A0.

El direccionamiento de las ROM, RAM y I-O Timer de los 7 chips, quedará de la siguiente forma:

		CS2 A12	CS1 A11	RSϕ A10	A9	A8	A7	A6
R6530 # 1,	ROM SELECT	0	0	1	X	X	X	X
	RAM SELECT	0	0	0	0	0	0	0
	I/O TIMER	0	0	0	1	0	0	0
R6530 # 2,	ROM SELECT	0	1	0	X	X	X	X
	RAM SELECT	0	0	0	0	0	0	1
	I/O TIMER	0	0	0	1	0	0	1
R6530 # 3,	ROM SELECT	0	1	1	X	X	X	X
	RAM SELECT	0	0	0	0	0	1	0
	I/O TIMER	0	0	0	1	0	1	0
R6530 # 4,	ROM SELECT	1	0	0	X	X	X	X
	RAM SELECT	0	0	0	0	0	1	1
	I/O TIMER	0	0	0	1	0	1	1
R6530 # 5,	ROM SELECT	1	0	1	X	X	X	X
	RAM SELECT	0	0	0	0	1	0	0
	I/O TIMER	0	0	0	1	1	0	0

		CS2 A12	CS1 A11	RSφ A10	A9	A8	A7	A6
R6530 # 6,	ROM SELECT	1	1	0	X	X	X	X
	RAM SELECT	0	0	0	0	1	0	1
	I/O TIMER	0	0	0	1	1	0	1
R6530 # 7,	ROM SELECT	1	1	1	X	X	X	X
	RAM SELECT	0	0	0	0	1	1	0
	I/O TIMER	0	0	0	1	1	1	0

De la tabla anterior se deduce que la selección de la memoria RAM del chip R 6530 # 5 será: $\overline{A12}.\overline{A11}.A10.\overline{A9}.A8.\overline{A7}.\overline{A6}$.

A continuación, se presenta otra tabla, que indica el direccionamiento de los diversos elementos del 6530. En ella se puede apreciar que la línea A2 sirve para distinguir cuándo se usa un registro I-O y cuándo el Timer o temporizador. Si A2 tiene nivel alto, selecciona el Timer y, cuando está bajo, los registros de I-O.

Cuando se ha seleccionado el Timer, las líneas A0 y A1 sirven para seleccionar el período (1T, 8T, 64T y 1.024 T).

A3 = 1 habilita IRQ a PB7 y si está a 0 lo inhibe.

	ROM Select	RAM Select	I/O Timer Select	R/W	A3	A2	A1	A0
Lectura ROM	1	0	0	1	X	X	X	X
Escritura RAM	0	1	0	0	X	X	X	X
Lectura RAM	0	1	0	1	X	X	X	X
Escritura DDRA	0	0	1	0	X	0	0	1
Lectura DDRA	0	0	1	1	X	0	0	1
Escritura DDRB	0	0	1	0	X	0	1	1
Lectura DDRB	0	0	1	1	X	0	1	1
Escritura Per. Reg. A	0	0	1	0	X	0	0	0
Lectura Per. Reg. A	0	0	1	1	X	0	0	0
Escritura Per. Reg. B	0	0	1	0	X	0	1	0
Lectura Per. Reg. B	0	0	1	1	X	0	1	0
Escritura Timer								
÷IT	0	0	1	0	·	1	0	0
÷8T	0	0	1	0	·	1	0	1
÷64T	0	0	1	0	·	1	1	0
÷1024T	0	0	1	0	·	1	1	1
Lectura Timer	0	0	1	1	·	1	X	0
Lectura Interrupt Flag	0	0	1	1	X	1	X	1

El Timer puede ser programado para contar hasta 255 intervalos de tiempo, cada uno de ellos de 1T, 8T, 64T o 1.024T, siendo T el período de la señal de reloj. Cuando se ha producido el contaje total, un flag de interrupción es puesto a 1. La máxima temporización que directamente puede proporcionar el TIMER será: 255 X 1.024 X T (1 μs) = 0,256 seg.

R 6500/1 SISTEMA MICROCOMPUTADOR EN UN SOLO CHIP

Como orientación, teniendo en cuenta la tendencia de los fabricantes en implementar sobre un sólo chip la mayor parte de los componentes de un sistema basado en microprocesador, describimos someramente el chip R 6500/1.

El R 6500/1 es un completo sistema microcomputador NMOS de 8 bits en un sólo chip. Es compatible con todos los miembros de la familia R 6500. Consta de los siguientes elementos:

CPU: R 6502.

ROM: 2 K bytes de 8 bits.

RAM: 64 bytes de 8 bits.

Contador programable de 16 bits y 4 modos de operación.

32 líneas de I-O (4 puertas A, B, C y D).

5 líneas de interrupción y una línea I-O contador. Las interrupciones son: Reset, NMI, 2 externas activas por flanco y contador.

En la figura 7-28 se muestra la arquitectura interna y el diagrama de pines del R 6500/1.

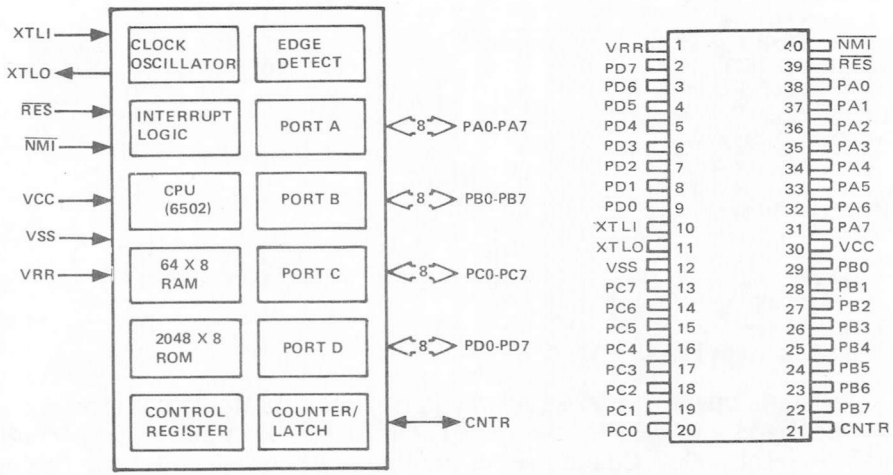


Fig. 7-28.–Arquitectura interna y diagrama de pines del 6500/1.

Teniendo en cuenta los últimos chips mencionados y su notable potencia, se pueden configurar sistemas completos de aplicación a base de microprocesador de 1 sólo chip con el R 6500/1, de interés en producciones muy importantes, puesto que se recibe de fábrica no sólo el sistema completo y ensamblado, sino también el programa grabado. Otra posibilidad, a tener en cuenta en producciones considerables, se tiene mediante el empleo del R 6530, que, junto con una CPU, constituyen el sistema básico alrededor de 2 chips. Finalmente, hay una 3.ª posibilidad para series cortas, utilizando el R 6532 junto con una CPU y una memoria PROM, programable por el propio usuario. En la figura 7-29 se muestra un resumen gráfico de las 3 posibilidades mencionadas con anterioridad.

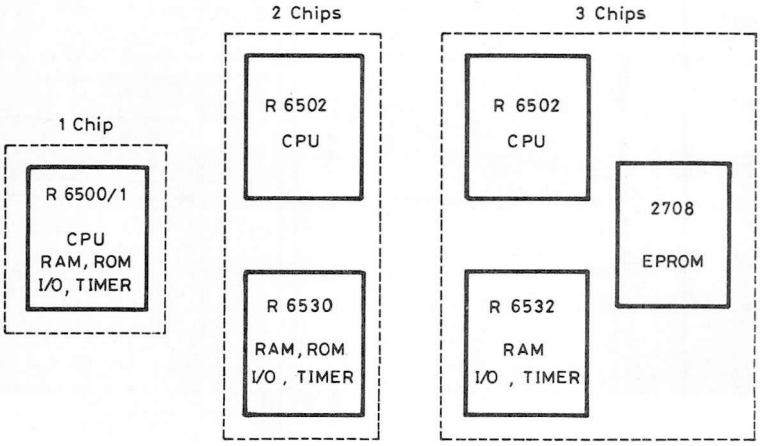


Fig. 7-29.–Representación gráfica de la implementación de un sistema básico microcomputador con 1, 2 ó 3 chips.

TARJETAS MODULARES PARA LA FAMILIA MICROCOMPUTADORA R 6500

Al igual que sucedía con la familia de componentes para el microprocesador de Motorola, también para la R 6500 existen diversas tarjetas de circuito impreso, ya montadas, que contienen conjuntos muy prácticos para facilitar la implementación de prototipos y series de fabricación. Exponemos algunos fabricados de COMELTA S. A., distribuidora de Rockwell en España.

Módulos de C.P.U. CR-101

Se trata de una tarjeta que consta de CPU (R 6502), R 6532 (RAM, I/O, TIMER), 2 VIAS (R 6522) y un conjunto de EPROM, modelo 2716 o 2708.

El conjunto dispone de 4 a 8 K de memoria de programa, 1157 bytes de datos, 56 pines de I/O seleccionables individualmente como entradas o salidas y 5 temporizadores o 2 contadores de impulsos, así como 4 pines de interrupción.

El diagrama por bloques de esta tarjeta se muestra en la figura 7-29/1.

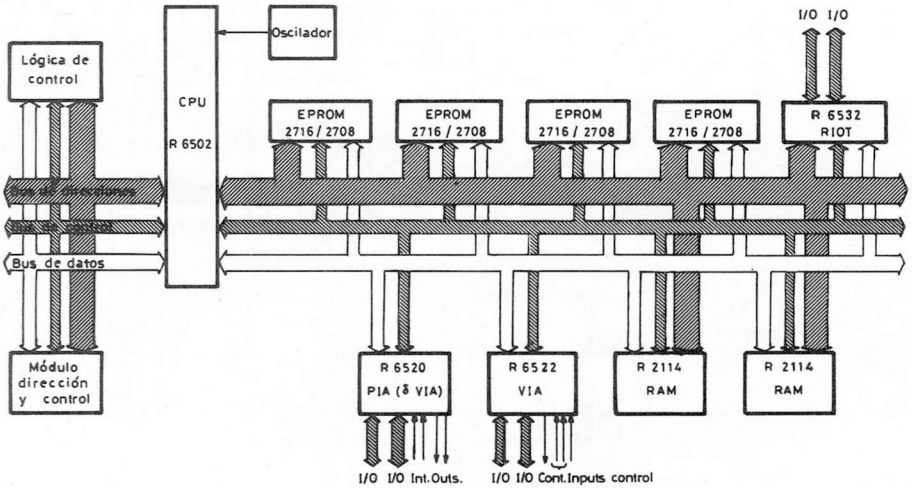


Fig. 7-29/1.–Diagrama por bloques de una tarjeta, que dispone un conjunto completo a base de un microprocesador, memoria y entradas y salidas.

La figura 7-29/2 muestra una fotografía de la tarjeta CR-101.

Otra tarjeta de aplicación a sistemas con microprocesador es la CR-106, módulo de RAM, que dispone de 8 K bytes de RAM estática, que ha sido diseñada para completar los prototipos y facilitar el hardware con garantía de funcionamiento. Usa 16 unidades RAM 2114 (1024 X 4 bits), según la estructura de la figura 7-29/3.

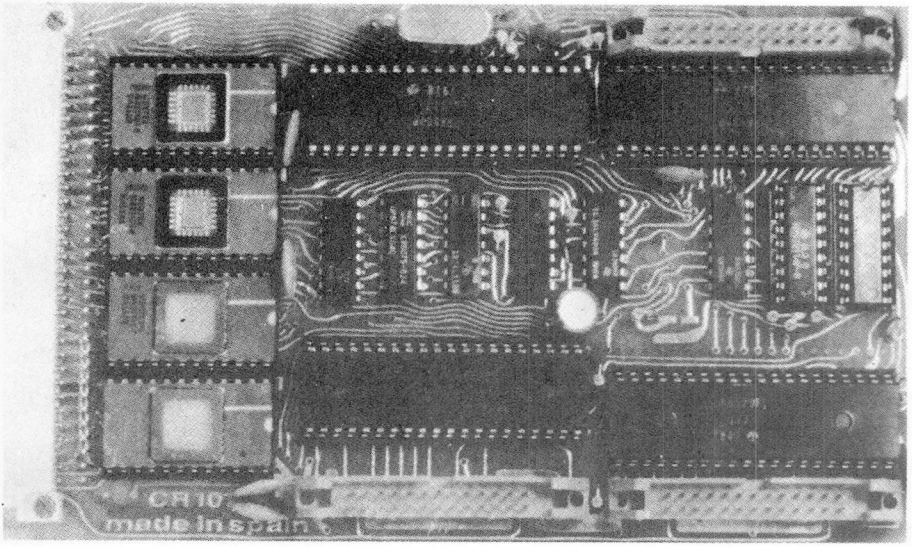


Fig. 7-29/2.–Fotografía de la tarjeta de módulo CPU.

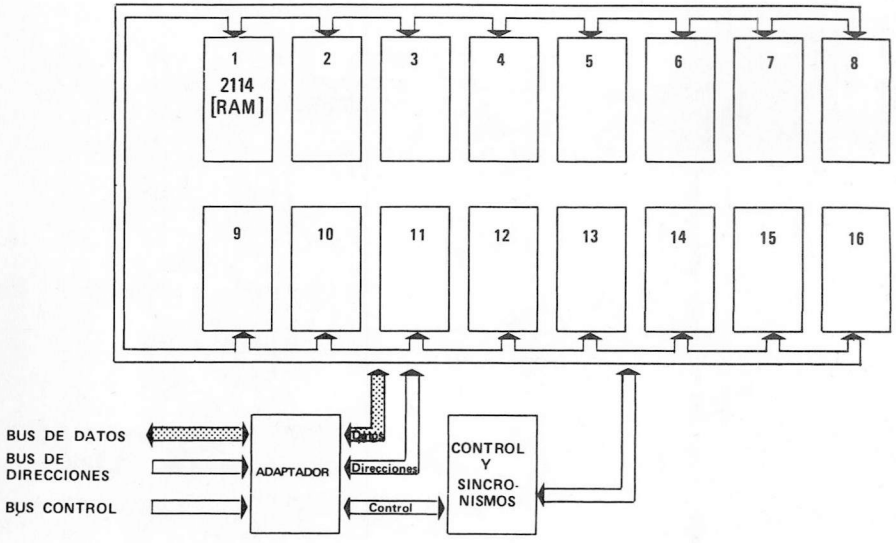


Fig. 7-29/3.–Diagrama por bloque de la tarjeta de RAM estática.

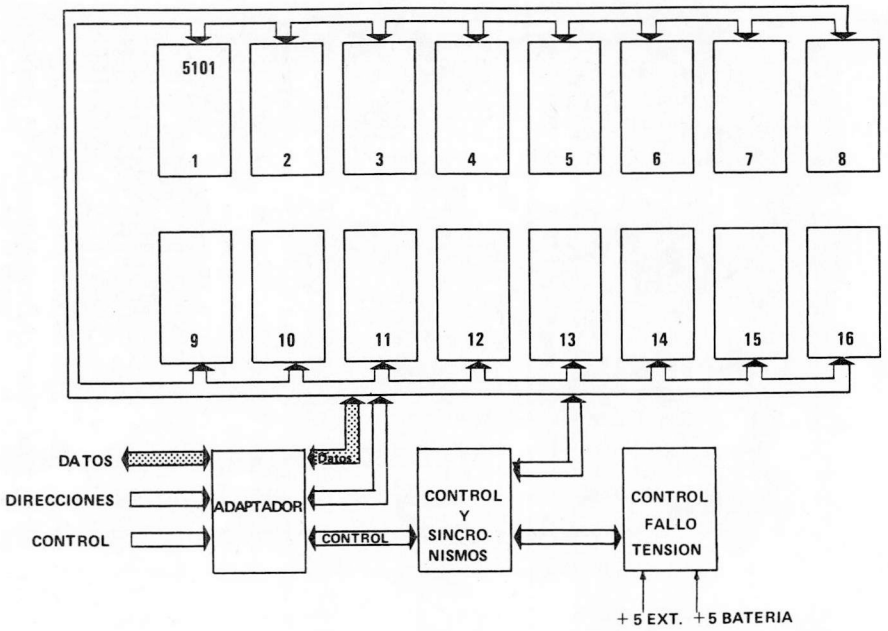


Fig. 7-29/4.–Diagrama por bloque del módulo de RAM C-MOS.

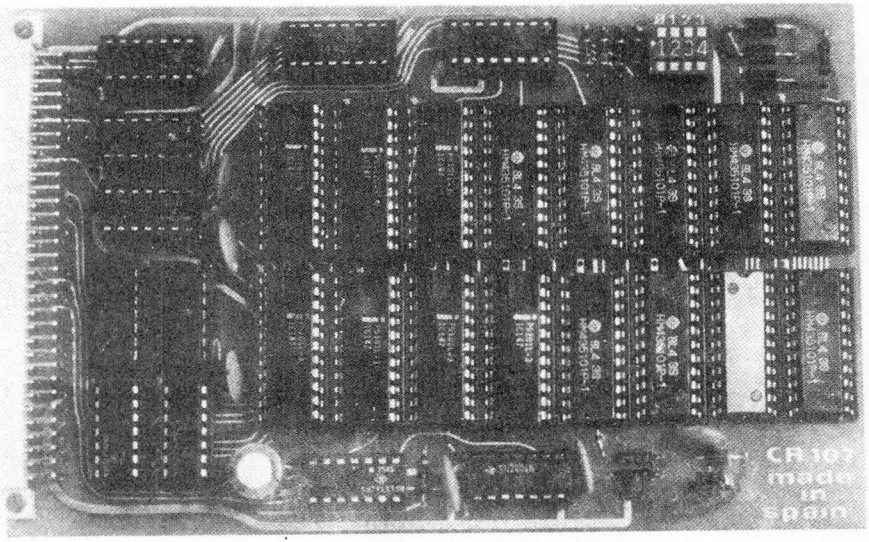


Fig. 7-29/5.–Fotografía del módulo CR-107 de RAM C-MOS.

En la figura 7-29/4 se ofrece el diagrama por bloques y en la 7-29/5 la fotografía de la tarjeta CR-107, módulo de RAM C-MOS con baterías. Se trata de un módulo que puede contener hasta 2 K bytes de memoria RAM C-MOS, capaz de aguantar, en caso de fallo la tensión, los datos almacenados hasta 500 horas con una batería Ni-Cd de 100 mA/h.

3.ª PARTE: FAMILIA MICROCOMPUTADOR PARA EL 8080 Y 8085 DE INTEL

Al igual que con las familias anteriores, se hará una descripción de los chips más representativos de la familia microcomputador para los microprocesadores INTEL.

GENERADOR DE CLOCK Y DRIVER PARA LA CPU 8080:8224

El 8224 es un generador de clock empleado en la base de tiempos, así como un driver para el microprocesador 8080. Recuérdese que el 8085 dispone el generador de clock integrado en el mismo chip.

El 8224 está controlado desde el exterior mediante un cristal, lo que permite obtener una gama de frecuencias ilimitada.

Asimismo, el 8224 incluye la lógica para la puesta a cero inicial de la CPU, la actuación de la palabra de estado del procesador y la sincronización READY para dispositivos de memoria y de entrada/salida más lentos que la CPU.

La selección de la frecuencia del cristal depende de la velocidad a que trabaje el 8080. La fórmula que define esta frecuencia es:

$$\text{Frecuencia del cristal} = \frac{1}{t_{CY}} \quad 9$$

Por ejemplo, si el período del 8080 es de 500 ns (t_{CY}), la frecuencia del cristal deberá ser de 18 MHz.

Las características del cristal a emplear son las siguientes: Tolerancia: 0,005 por 100 entre 0-70°. Resonancia: serie. Carga capacitiva: 20-35 pF. Resistencia equivalente: 75-20 ohmios. Potencia mínima de disipación: 4 mw.

En la figura 7-30 se muestran el diagrama del conexionado de los pines del 8224 y el esquema por bloques del mismo.

La misión de las diferentes entradas y salidas de este chip son las siguientes:

Señales de entrada

XTAL1 y XTAL2: Conexiones para el cristal externo.

TANK: Se usa cuando el cristal empleado es con resonancia en un armónico.

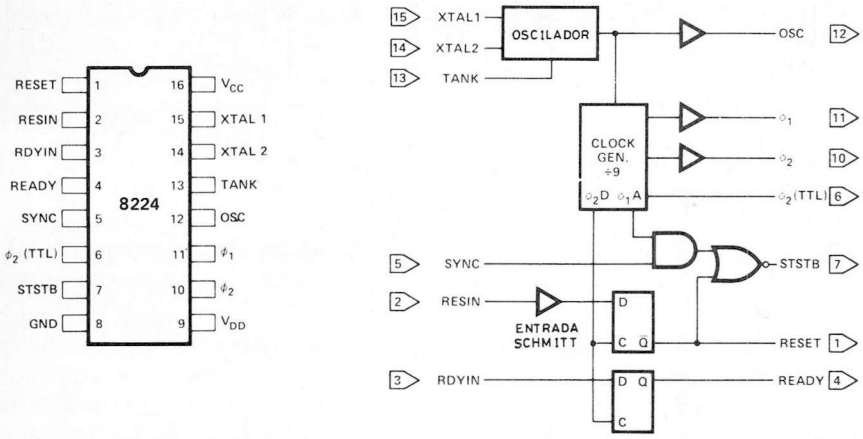


Fig. 7-30.–Diagrama de conexionado y esquema por bloque del 8224.

SYNC: Entrada de sincronismo, que genera la CPU 8080 y que aparece al principio de cada ciclo máquina.

$\overline{\text{RESIN}}$: Entrada procedente del pulsador exterior de puesta a cero.

RDYIN: Señal procedente de un dispositivo de memoria o de entrada y salida, en la que se definen si los datos están ya colocados en el bus o no.

Señales de salida

OSC: Salida del oscilador para crear una base de tiempos externa.

ϕ_1 y ϕ_2: Fases de reloj para el 8080.

ϕ_2 (TTL): Fase de reloj 2 a nivel TTL.

$\overline{\text{STSTB}}$: Se emplea para actuar la palabra de estado del procesador.

RESET: Indica al procesador una operación de puesta a cero.

READY: Indica a la CPU si los datos de los dispositivos de memoria o de entrada/salida están ya en el bus de datos.

Vcc: +5V ± 5 por 100.

GND: Tierra.

En una figura posterior se ofrecerá el conexionado del 8224 con la CPU 8080.

CONTROLADOR DEL SISTEMA Y DRIVER DEL BUS PARA EL 8080: 8228

El 8228 es un chip que contiene un controlador del sistema y un driver del bus de datos del 8080, que forma parte de la familia microcomputador, denominada MCS-80.

El driver del bus bidireccional proporciona un sistema compatible con TTL y con una elevada capacidad de carga a la salida. Asimismo, proporciona el aislamiento del bus de datos del 8080 de los componentes de memoria y de entrada/salida, lo cual optimiza las señales de control, permitiendo el diseño de sistemas con memorias y elementos de entrada/salida más lentos. Además el aislamiento del bus proporciona una mayor inmunidad al ruido.

El 8228 está diseñado para soporte de una variedad de estructuras del bus del sistema, reduciendo el tamaño, número de componentes y coste en el diseño de sistemas MCS-80.

La figura 7-31 ofrece el diagrama de conexionado y el esquema por bloques de 8228.

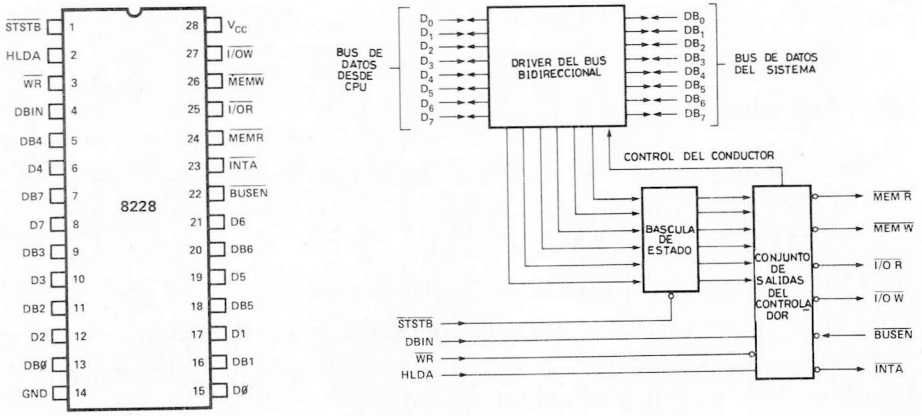


Fig. 7-31.–Diagrama y esquema por bloque del 8228.

Se expone a continuación una somera descripción de las señales, que entran o salen del 8228:

Señales de entrada

$\overline{\text{STSTB}}$: Esta señal, procedente de un 8224, permite el almacenamiento en una báscula interna de la palabra de estado del 8080, que define las operaciones a realizar una vez decodificada. Ver figura 7-33.

HLDA: Señal que procede del 8080 y que indica que el procesador se encuentra en un estado HOLD.

DBIN: Señal procedente del 8080. Indica que el bus de datos del 8080 está en modo de entrada.

D0-D7: Contenido del bus de datos de la CPU, que debe enviarse al bus de datos del sistema.

$\overline{\text{BUSEN}}$: Obliga a una elevada impedancia en las salidas del bus de datos y señales de control (tri-estado).

Señales de salida

DB0-DB7: Contenido del bus de datos del sistema.

$\overline{\text{MEM R}}$: Indica una operación de lectura de memoria.

$\overline{\text{MEM W}}$: Indica una operación de escritura de memoria.

$\overline{\text{I/O R}}$: Indica una operación de lectura en un dispositivo de entrada.

$\overline{\text{I/O W}}$: Indica una operación de escritura en un dispositivo de salida.

$\overline{\text{INTA}}$: Si esta señal se conecta a +12 V a través de una resistencia de 1 K, el 8228 proporciona automáticamente una instrucción RST 7, cada vez que hay una demanda de interrupción. Normalmente se utiliza para colocar en el bus la instrucción de interrupción.

Vcc: +5 V ± 5 por 100.

GND: Tierra.

En la figura 7-32 se ofrece un detalle del conexionado entre el 8228 y la CPU.

En la figura 7-34 se ofrece el circuito en que se relaciona la CPU 8080 con los 2 chips descritos hasta el momento.

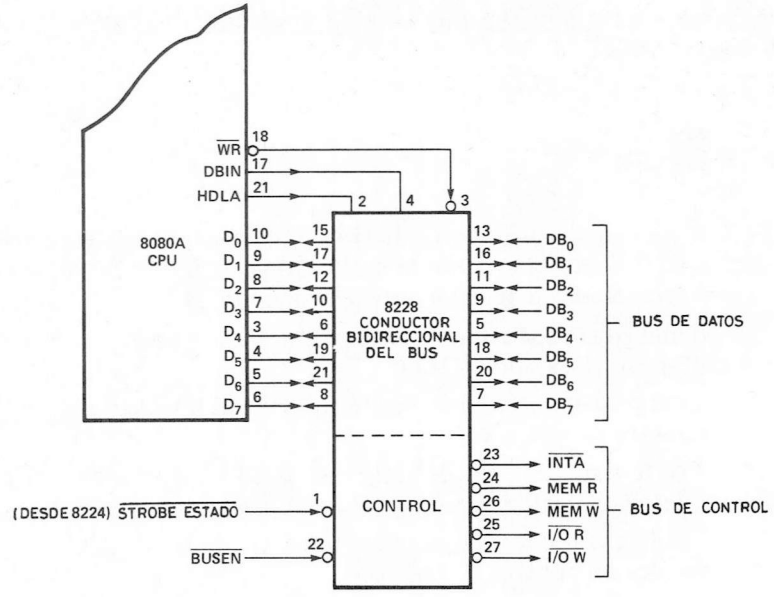


Fig 7-32.–Detalle del conexionado entre el 8228 y el 8080.

MEMORIA REPROM DE 2.048 BITS. 8720 A

La memoria 8702 A es una ROM programable eléctricamente con una capacidad de 256 palabras de 8 bits, recomendada especialmente para el desarrollo de sistemas de microcomputador, debido a la facilidad de manejo y de modificación del contenido. Permite programar individualmente cada uno de los 2.048 bits, lo cual supone un 100 por 100 de programación.

Está implementada en una cápsula de 24 patitas, dual-in-line, con una tapa de cuarzo transparente, que permite al usuario exponer el circuito a rayos ultravioleta para borrar el contenido, pudiéndose con posterioridad grabar una nueva información.

Sus características más relevantes son:

Tiempo de acceso: 1,3 μs máximo.

Velocidad de programación: 2 minutos para los 2 K bits.
Tecnología de fabricación: MOS estática.
Entradas y salidas compatibles con TTL.
Salida tri-estado.

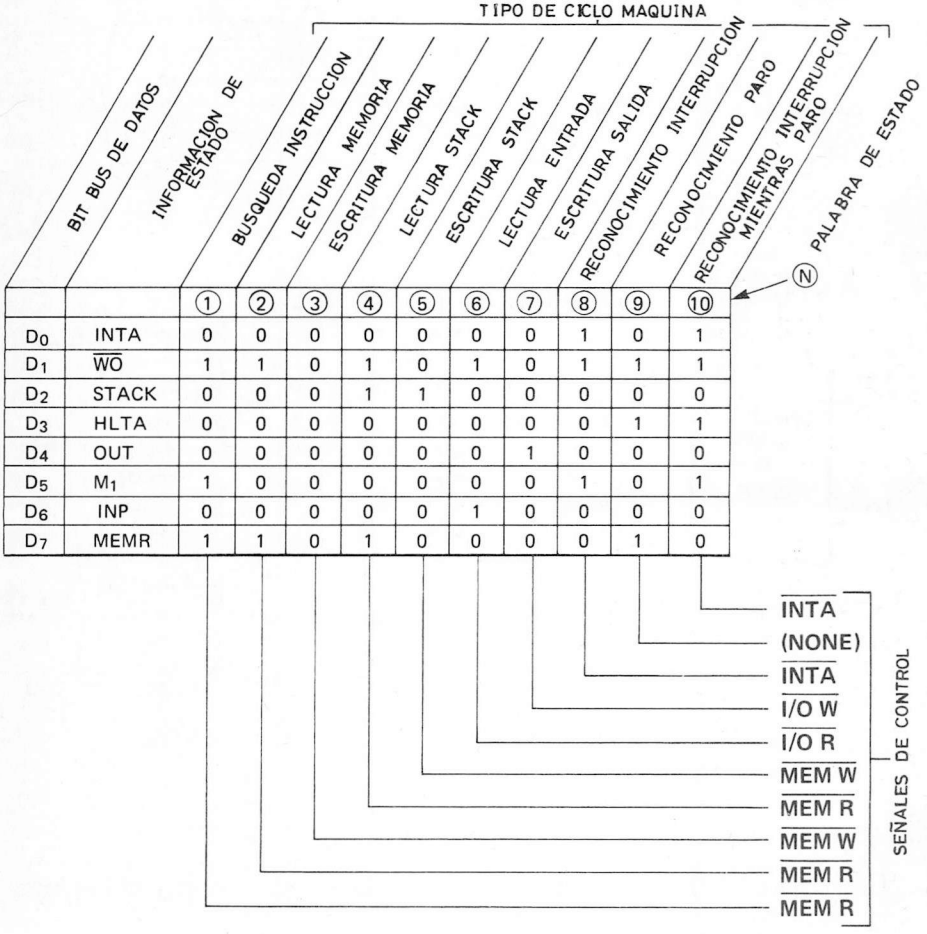


Fig. 7-33.–Diagrama de la palabra de estado.

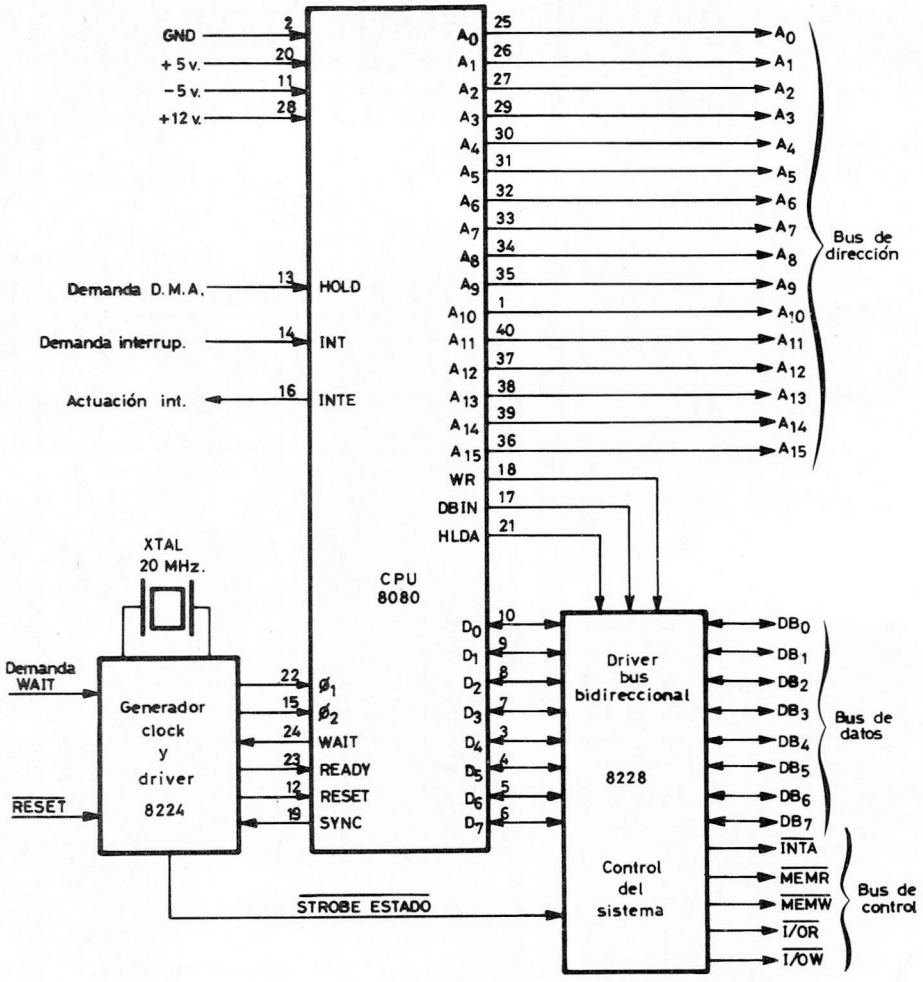


Fig. 7-34.–Circuito de conexionado del 8080 con el 8224 y el 8228.

En la figura 7-35 se muestra el diagrama de conexionado y el esquema por bloques de esta memoria.

La descripción de los pines de la 8702 A es la siguiente:

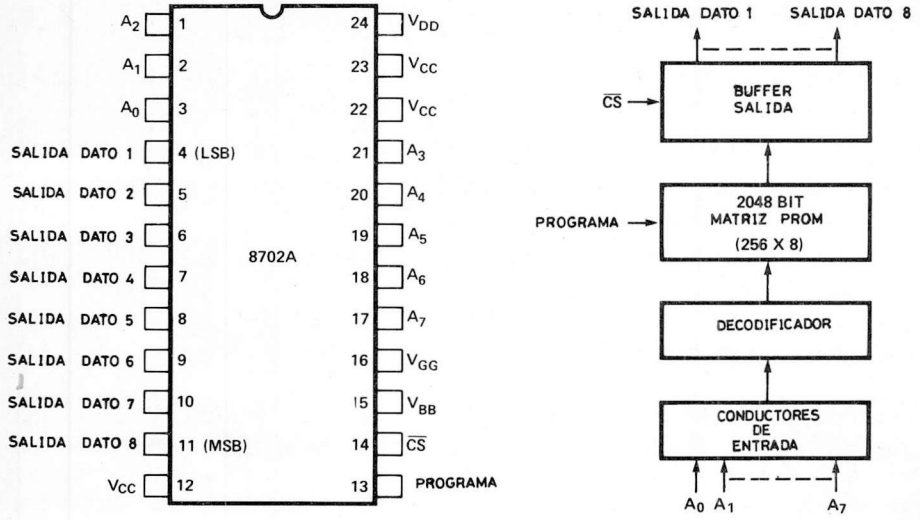


Fig. 7-35.–Diagrama de conexionado y esquema por bloques de la memoria REPROM 8702 A.

Señales de entrada

A7-A0: Mediante estos 8 bits se selecciona una de las 256 palabras de 8 bits del interior de la memoria.

$\overline{CS}$: Selección de chip. Permite seleccionar o inhibir un chip. Cuando está inhibido sus salidas de datos tienen una elevada impedancia (tri-estado).

PROGRAMA: Aparece esta señal cuando se procede a la programación eléctrica del chip.

Señales de salida

DO7-DOϕ: Salidas de datos tri-estado. Aparece el contenido de una de las 256 palabras de la memoria.

Alimentación: V_{cc}: +5 V ± 5 por 100; V_{dd}: −9 V ± 5 por 100; V_{gg}: −9 V ± 5 por 100.

Condiciones generales

Temperatura: de trabajo: 0-70° C; de almacenamiento: −65 a 125° C y de soldadura: 300° C.

Potencia de disipación: 2 watios.

Operación lectura: Tensión de entrada y alimentación respecto a V_{cc}: +0,5 a −20 V.

Operación de programación: Tensión de entrada y alimentación respecto a V_{cc}: −48 V.

Programación de PROM

Intel proporciona sistemas de desarrollo de programas, Intellec, que pueden utilizarse para la programación de sus PROMs. Para ello van equipados con una tarjeta "programador de PROM", que se utiliza con las memorias ROM programables eléctricamente y borrables con ultravioleta.

Como dispositivo de entrada se emplea un teletipo y los programas a grabar en las PROMs pueden introducirse directamente o ser cargados mediante el lector de la cinta.

También existen equipos específicos para la grabación de las PROM.

Borrado de la PROM

La 8702 A puede borrarse totalmente mediante su exposición a una elevada intensidad de luz ultravioleta de 2.537 Å. La dosis de exposición recomendada es de 6 W/cm^2.

Las lámparas deben utilizarse sin filtros, colocando la 8702 A a una distancia del tubo de 2 centímetros aproximadamente.

REPROM 8708

Otro chip de memoria similar al anterior es el 8708, cuyas características más destacables son:

Tiempo de acceso: 450 ns.

Tiempo de programación: 100 segundos en total.

Consta de 1.024 palabras de 8 bits.

Entradas y salidas compatibles con TTL. Salidas tri-estado.

Puede utilizarse en lugar de la memoria ROM 8308, programable por máscara.

El diagrama de conexionado y el esquema por bloques se ofrece en la figura 7-36.

La alimentación necesaria para esta memoria es: V_{ss}: 0; V_{bb}: -5 V ± ± 5 por 100; V_{dd}: +12 V ± 5 por 100; V_{cc}: +5 V ± 5 por 100.

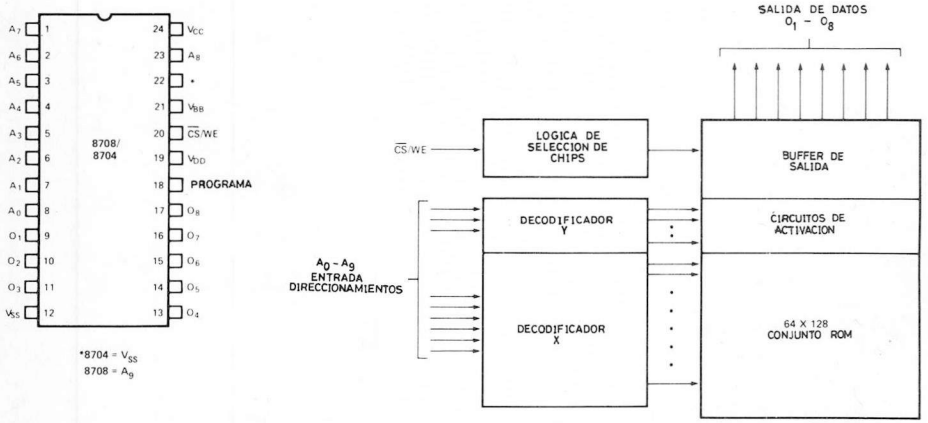


Fig. 7-36.—Configuración de las patitas y esquema por bloque de la memoria REPROM 8708.

MEMORIA RAM ESTATICA 8111-2

La Intel 8111-2 es una memoria de acceso aleatorio de 256 palabras de 4 bits, fabricada con tecnología MOS canal N-estática, por lo que no necesita señales de reloj ni de refresco. La lectura de datos no es destructiva.

La 8111-2 está recomendada para aplicaciones de memoria en pequeños sistemas, por su bajo coste, gran capacidad de almacenamiento y fácil interconexión.

Las características más importantes son:

Tiempo de acceso: 850 ns máximo.

Entrada y salida de datos común.

Unica alimentación de +5 V.

Salidas tri-estado y bajo consumo de unos 150 mw.

La figura 7-37 muestra los aspectos más representativos de esta memoria.

Las líneas A7-A0 son de direccionamiento y por las I/O1 - I/O4 se introducen o se sacan los datos de las posiciones de memoria.

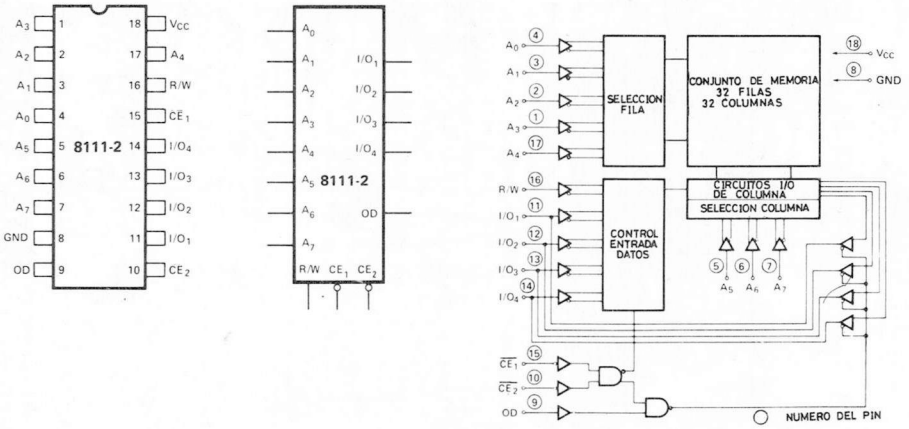


Fig. 7-37.–Configuración de pines y esquema por bloques de la memoria RAM 8111-2.

La línea OD inhibe las salidas colocando en las mismas una elevada impedancia. $\overline{CE1}$-$\overline{CE2}$ seleccionan al chip tanto en la aceptación como en la salida de datos. R/W, determina si es lectura o escritura.

La figura 7-37 (2) muestra una supuesta memoria RAM de sólo 8 células de almacenamiento, representadas por flip-flops D, las cuales están conectadas a una sola línea bidireccional de entrada y salida de datos del tipo triestado. En general, las memorias RAM responden a esta estructura.

PUESTA DE ENTRADA Y SALIDA DE 8 BITS: 8212

La puerta de entrada/salida 8212 es una báscula de 8 bits con salida tri-estado, mediante buffers, además de la lógica de control y selección de dispositivo. Incluye un flip-flop de demanda de servicio para la generación y control de interrupciones.

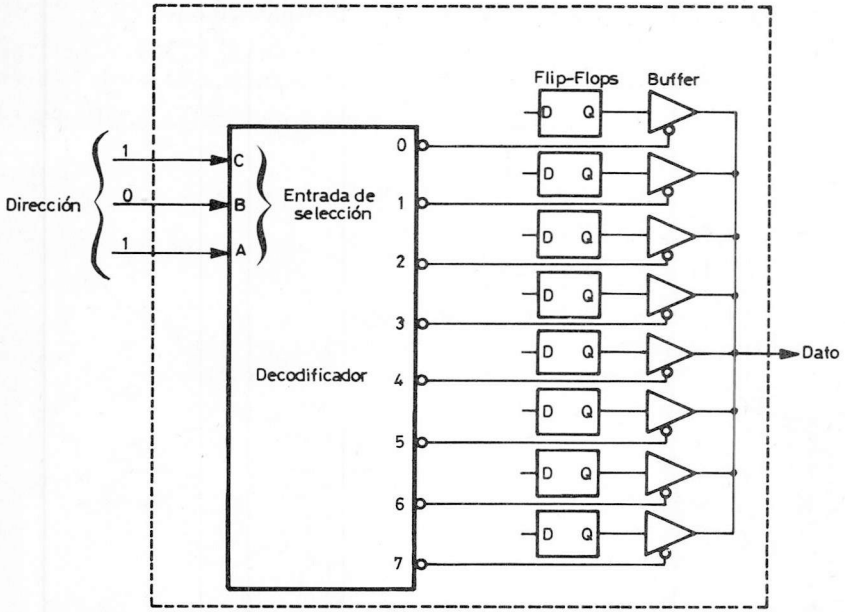


Fig. 7-37 (2).–Memoria RAM de 8 bits (Flip-flops) con una linea bidireccional de datos. La célula leida en la figura es la 5ª.

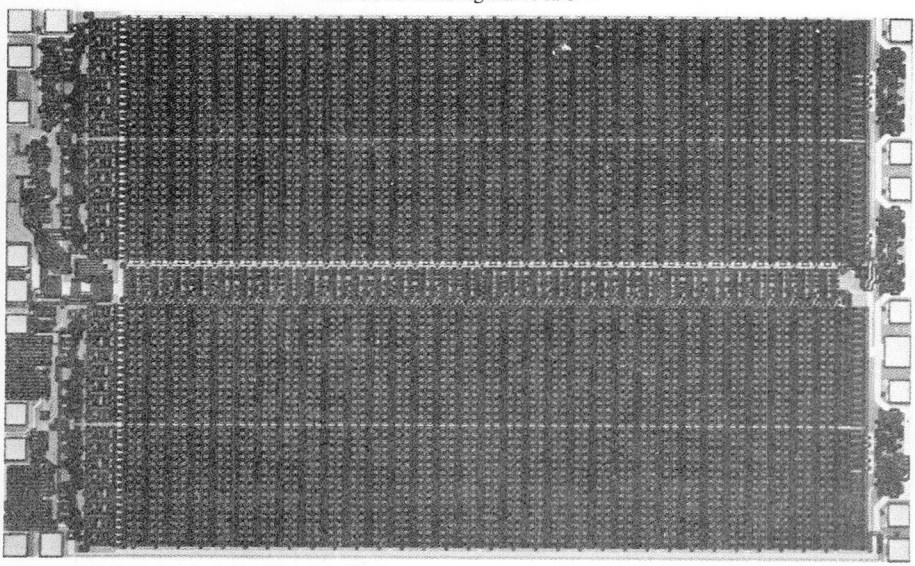


Fig. 7-37 (3).– Fotomicrografía de un chip de 0,225 X 0,142 pulgadas, que constituyen una memoria RAM 4.096 bits. Cortesía de INTEL.

Las posibles aplicaciones de este chip son las siguientes:

- Como buffer simple, activado por las señales DS1 y DS2. Cuando no se le selecciona, pone una elevada impedancia a su salida.
- Como driver de bus direccional, cableando adecuadamente dos circuitos 8212.
- Como puerta de entrada de interrupción.
- Como puerta de instrucción de interrupción.
- Como puerta de salida tri-estado.
- Como báscula de estado del 8080.

En la figura 7-38 se muestran algunos detalles del 8212.

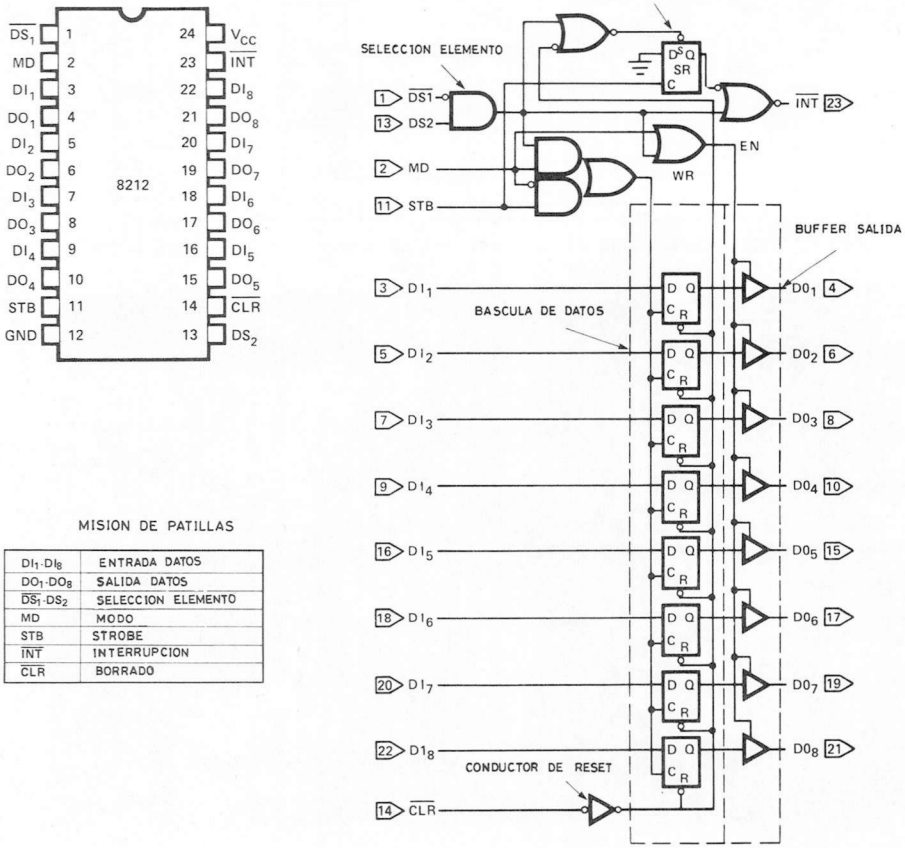


Fig. 7-38.–Configuración y esquema por bloque del 8212.

Señales de entrada

DI1-DI8: Entrada de datos.

$\overline{DS1}$-DS2: Selección del circuito. Si no está seleccionado, salida tri-estado.

MD: Modo. Controla el estado del buffer de salida y determina la procedenteic adel clock de las básculas de datos internas.

STB: Strobe. Se usa como clock de las básculas de datos.

$\overline{CLR}$: Borra las básculas de datos cuando es "0".

Señales de salida

$\overline{INT}$: Entrada en los circuitos de generación de prioridad de interrupción.

D01-D08: Salida de datos (buffer).

Alimentación: V_{cc}: + 5 V ± 5 por 100. GND: tierra.

En la figura 7-39 se muestra al 8212 como elemento de salida y como elemento de entrada.

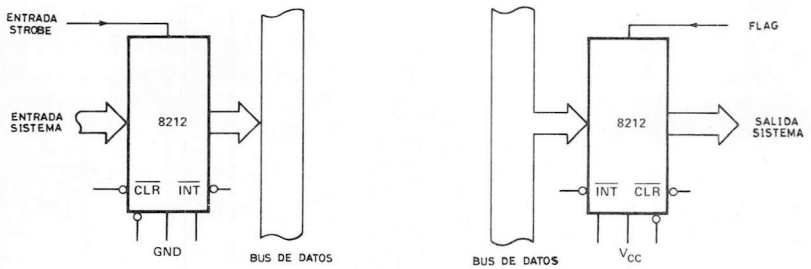


Fig. 7-39.–Esquemas del 8212 usado como elemento de entrada y salida.

INTERFACE PROGRAMABLE PARA PERIFERICOS: 8255

El 8255 es un dispositivo de entrada/salida programable, diseñado para su utilización con los microprocesadores Intel 8008, 8080 y 8085.

Tiene 24 terminales de entrada/salida, que pueden ser programados individualmente en dos grupos de doce y utilizados según 3 modos básicos

de operación. Es totalmente compatible con TTL y con los elementos de la familia microcomputador de Intel MCS-8 y MCS-80.

Otras posibilidades del 8255 son las de poner a uno o a cero los bits directamente y proporcionar una corriente de 1 mA a una tensión de 1,5 V. Ello permite conectar directamente transistores Darlington para aplicaciones a impresoras y displays de alta tensión.

En la figura 7-40 se muestran detalles del 8255.

En la figura 7-40 aparece la descripción de los terminales del 8255, que es la siguiente:

Señales de entrada

RESET: Puesta a cero. Borra todos los registros internos, incluso el de control, colocando en modo de entrada a todas las puertas.

$\overline{CS}$: Selección de circuito. Permite o inhibe la comunicación entre el 8080 y el 8255.

$\overline{RD}$: Lectura. Cuando está a cero permite leer a la CPU del 8255.

$\overline{WR}$: Escritura.

A0-A1: Selección de puerta 0 y de puerta 1. Estas señales, en combinación con RD y WR, seleccionan una de las tres puertas o el registro de Palabra de Control. Se suelen conectar a A0-A1 de direccionamiento.

D7-D0: Transferencia de la información del bus de datos bidireccional.

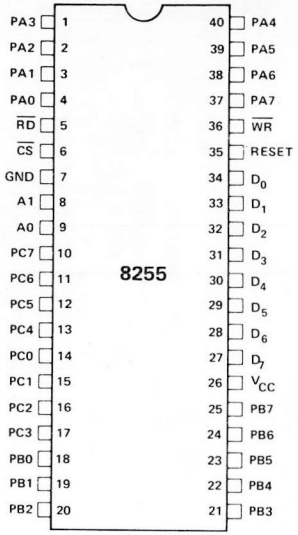
Alimentación: V_{cc}: + 5 V ± 5 por 100. GND: tierra.

Señales de entrada/salida

PA7-PA0: Buffer de salida de datos y báscula de entrada.

PB7-PB0: Buffer de salida de datos y báscula de entrada.

PC7-PC0: Buffer de salida de datos y báscula de entrada. Esta puerta puede dividirse en dos puertas de 4 bits, según el modo de trabajo. Cada puerta de 4 bits contiene una báscula de 4 bits y puede utilizarse para las salidas de las señales de control y entrada de las señales de estado en combinación con la puerta A y B.



MISION DE PATILLAS

D_7–D_0	BUS DE DATOS (BIDIRECCIONAL)
RESET	ENTRADA RESET
$\overline{CS}$	SELECCION CHIPS
$\overline{RD}$	ENTRADA LECTURA
$\overline{WR}$	ENTRADA ESCRITURA
A0, A1	DIRECCION PUERTA
PA7-PA0	PUERTA A (BIT)
PB7-PB0	PUERTA B (BIT)
PC7-PC0	PUERTA C (BIT)
V_{CC}	+5 v.
GND	0 v.

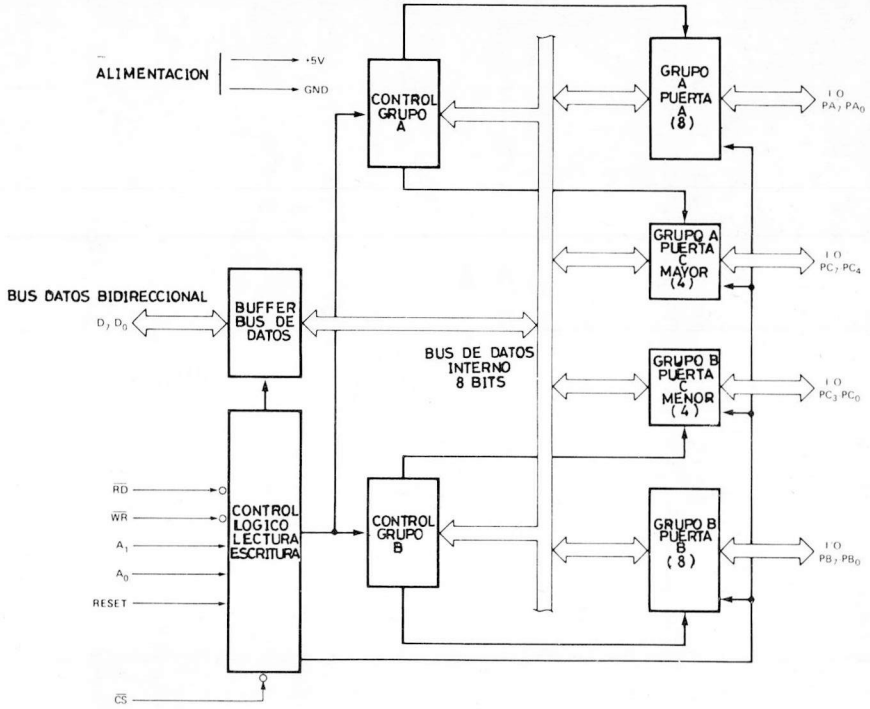


Fig. 7-40.–Configuración y esquema por bloques del 8255.

Modos de funcionamiento del 8255

MODO ϕ. Entrada y salida básica

Esta configuración permite operaciones simples de entrada y salida para cada una de las 3 puertas. En este modo no se precisan bits de control y los datos simplemente se escriben o se leen de una puerta determinada.

Sus características son:

a) Cualquier puerta puede ser de entrada y salida.

b) Se pueden configurar 2 puertas de 8 bits y 2 de 4 bits.

c) Las salidas contienen básculas pero las entradas son directas.

d) Existen 16 posibles configuraciones de entrada/salida.

MODO 1

Esta configuración proporciona el medio de transferir datos de entrada/salida a o desde una puerta determinada en combinación con palabras de control y diálogo. En este modo la puerta A y la puerta B utilizan las líneas de la puerta C para generar o aceptar las señales de diálogo.

Este modo se caracteriza por:

a) Existir dos puntos: A y B, cada uno de los cuales dispone de una puerta de 8 bits y una puerta de 4 bits de control/datos.

b) Las puertas de datos de 8 bits pueden ser ambas de entrada o salida mediante básculas en los dos casos.

c) Las puertas de 4 bits se usan para control y estado de las puertas de 8 bits.

MODO 2

Esta configuración permite la comunicación con un dispositivo periférico o una estructura con un bus de 8 bits bidireccional. Las señales de control permiten controlar el flujo de datos sobre el bus de datos, de forma similar al modo 1. También contiene la generación de interrupciones y funciones de actuación/inhibición. Características:

a) Sólo se usa el grupo A.

b) Una puerta A como bus de datos bidireccional de 8 bits y una puerta de control C con 5 bits.

c) La puerta de control de 5 bits C se emplea como control y estado del bus bidireccional. Las entradas y salidas son con básculas.

En la figura 7-41 se muestra un esquema sinóptico de la actuación de los 3 modos de funcionamiento del 8255.

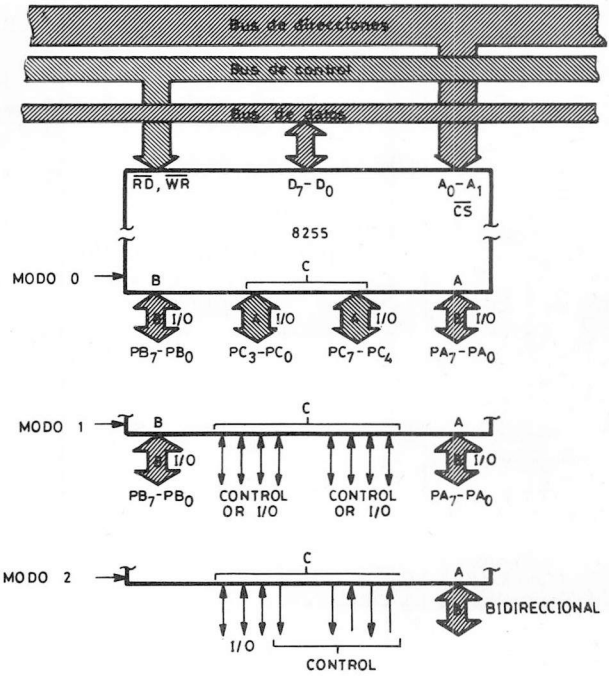
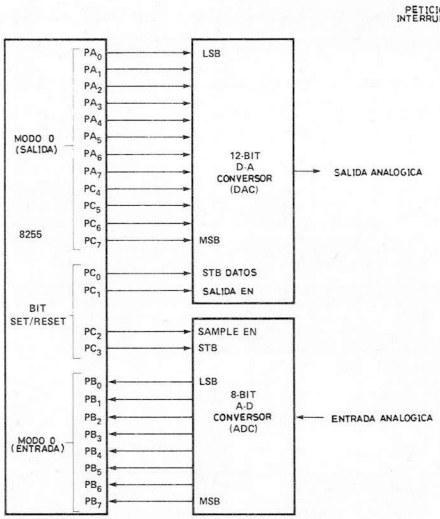


Fig. 7-41.—Actuación de los 3 modos de funcionamiento del 8255.

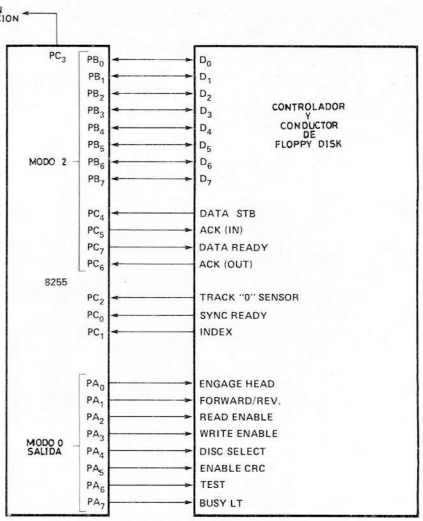
La figura 7-42 muestra la actuación del 8255 en los 3 modos posibles de funcionamiento. En el modo 0, como un elemento conversor de señales analógicas en digitales y viceversa; en el modo 1, como un adaptador a un CRT y en el modo 2, como un interface a un sistema de Floppy Disc.

INTERFACE PROGRAMABLE DE COMUNICACION 8251 (USART)

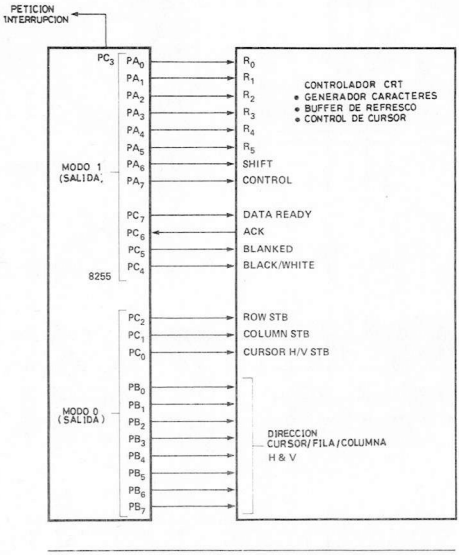
El 8251 es un Receptor/Emisor Síncrono/Asíncrono Universal (USART), diseñado para la transmisión de datos en sistemas de micro-



Digital a Analógico, Analógico a Digital



Interface básico con Floppy Disc



Controlador para Interface básico con CRT

Fig. 7-42.–Aplicaciones de los 3 modos de funcionamiento del 8255.

computador. Se usa como un dispositivo periférico y se programa por la CPU para utilizar cualquier técnica de transmisión de datos en serie.

La USART acepta caracteres de datos de la CPU en formato paralelo y los convierte en datos en serie para la transmisión. Simultáneamente puede recibir datos en serie y convertirlos para la CPU en paralelo. La USART indica a la CPU cuándo puede aceptar o cuándo puede recibir un carácter de ésta. La CPU puede leer el estado completo de la USART en cualquier instante. Este estado incluye errores en la transmisión de datos y señales de control.

Las características fundamentales del 8251 son:

Operaciones síncronas: Caracteres de 5 a 8 bits.
Sincronización interna o externa.
Inserción automática de sincronismo.

Operaciones asíncronas: Caracteres de 5 a 8 bits.

Clock de transmisión 1, 16 o 64 veces la velocidad de transmisión en baudios.
Generación de carácter de ruptura de transmisión 1, 1,5 o 2 bits de stop.
Detección de bit de puesta en marcha erróneo.

Velocidad de transmisión de 56 K baudios en modo síncrono y de 9,6 k Baudios en modo asíncrono.

Detección de error de paridad y formato.

Compatible con 8080. Unica alimentación a + 5 V y único clock TTL.

Las características más importantes del circuito, construido con tecnología canal N silicon-gate, se muestran en la figura 7-43.

Señales de entrada

C/$\overline{D}$: Esta señal, en combinación con WR y RD, informa al 8521 de que la palabra colocada en el bus de datos es un carácter de datos, una palabra de control o información de estado.

$\overline{RD}$: Lectura. Cuando es 0 indica que la CPU lee información en el 8251.

$\overline{WR}$: Escritura.

$\overline{CS}$: Selección de circuito.

CLK: Clock. La entrada CLK genera las dos temporizaciones del 8251. Suele estar conectada a ϕ_2 (TTL) del generador 8224.

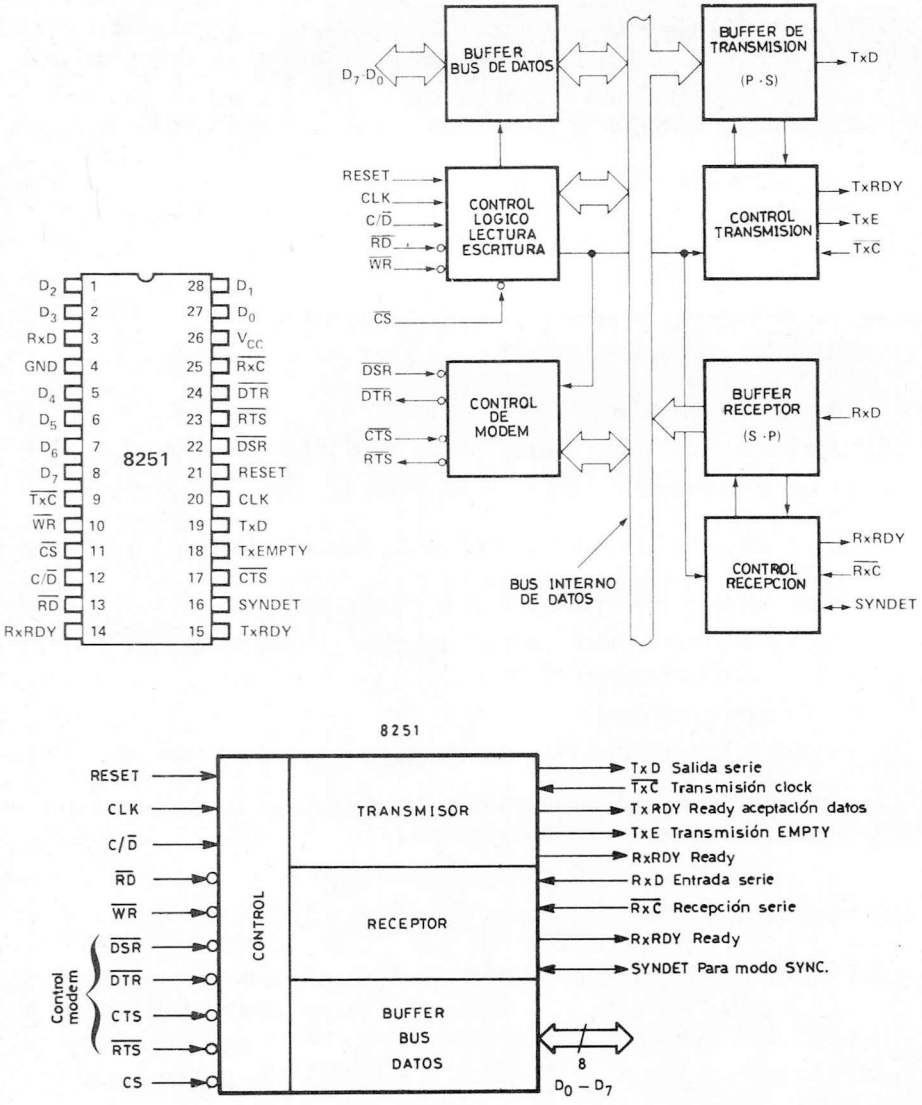


Fig. 7-43.–Configuración y esquema por bloque del 8251.

RESET: Puesta a cero. Inhibe al 8251 hasta que se le envía otra palabra de control.

DSR: Se utiliza en la transmisión por MODEM.

$\overline{CTS}$: Se utiliza en la transmisión por MODEM.

$\overline{TxC}$: Clock de transmisión. Controla la velocidad a que se transmite el carácter.

RxD: Entrada de datos en serie.

$\overline{RxC}$: Clock de recepción. Controla la velocidad a que se recibe el carácter.

Señales de salida

$\overline{DTR}$: Se utiliza en la transmisión por MODEM.

$\overline{RTS}$: Se utiliza en la transmisión por MODEM.

RxRDY: Receptor listo. Indica que el 8251 está preparado para enviar un carácter a la CPU.

TxE: Cuando el 8251 no tiene caracteres para transmitir esta señal está a 1. Cuando recibe un carácter de la CPU pasa a 0 automáticamente.

TxRDY: Indica a la CPU que el 8251 está preparado para recibir un carácter de la CPU.

TxD: Salida de datos en serie.

Señales de entrada/salida

D7-D0: Líneas del bus de datos bidireccional.

SYNDET: Detección de sincronismo. Sólo se usa en modo síncrono.

Alimentación: V_{cc}: + 5 V ± 5 por 100. GND: tierra.

Finalmente, en la figura 7-44 se muestra un detalle del circuito de interfase del 8251 con el sistema de buses del 8080.

El direccionamiento del 8251 se muestra en el siguiente cuadro

C/D	RD	WR	CS	OPERACION
0	0	1	0	Lectura del 8251 al bus de datos.
0	1	0	0	Escritura del bus de datos al 8251.
1	0	1	0	Estado al bus de datos.
1	1	0	0	Bus de datos a control.
–	–	–	1	Bus de datos a triestado.

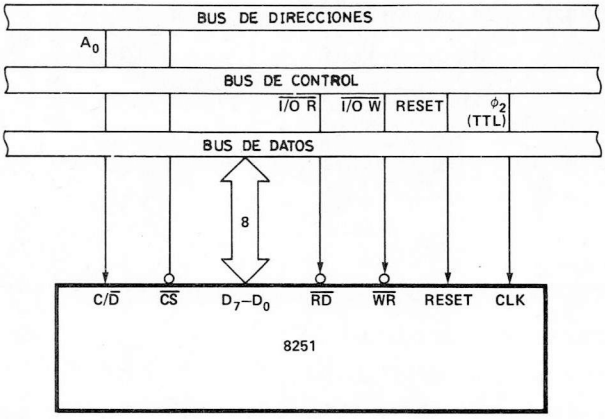


Fig. 7-44.–Interface del 8251 con los buses del sistema 8080.

DRIVER DE BUS BIDIRECCIONAL DE 4 BITS EN PARALELO 8216

El 8216 es un driver de bus bidireccional de 4 bits, diseñado específicamente como buffer de los componentes de un sistema microcomputador.

Todas las entradas son compatibles con TTL, en tanto que las salidas D0, como quiera que van conectadas generalmente a circuitos MOS, como es el caso de la CPU 8080, dan una tensión de 3,65 V, que permite la conexión directa con esta unidad central de proceso, cuya entrada mínima es de 3,3 V, lo cual da un margen de ruido de 350 mV.

Por otro lado, las salidas DB, al alimentar estructuras relativamente complejas, proporcionan una corriente de carga a la salida de 50 mA. También dispone de una baja corriente de entrada (0,25 mA máx.) y salias tri-estado.

Asimismo, mediante la utilización del 8216, aparte de reducirse el número de componentes, se obtiene una inmunidad al ruido y un aislamiento del bus de datos de la CPU realmente aconsejables.

La figura 7-45 muestra algunas características importantes de la arquitectura de este circuito.

Señales de entrada del 8216

DI0-DI3: Datos de entrada procedentes del bus de datos de la CPU.

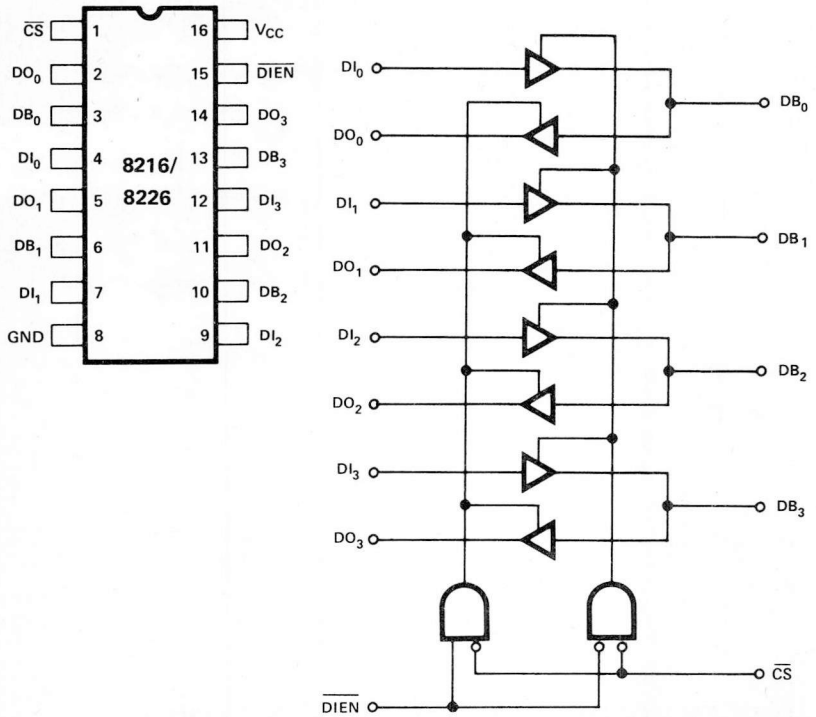


Fig. 7-45.—Configuración y diagrama por bloque del 8216.

$\overline{\text{DIEN}}$: Determina la dirección del flujo de datos a través de las líneas bidireccionales. Coloca a la salida una elevada impedancia (tri-estado) cuando el circuito no está seleccionado.

$\overline{\text{CS}}$: Selección de circuito.

Señales de salida

D01-D03: Datos de salida hacia el bus de datos de la CPU.

DB1-DB3: Líneas bidireccionales que constituyen el bus de datos externos.

Alimentación: + 5 V ± 5 por 100. GND: tierra.

Por último, se muestran dos esquemas sobre la adaptación del 8216 al bus de datos de la CPU y a elementos de memoria o de I/O. Figura 7-46.

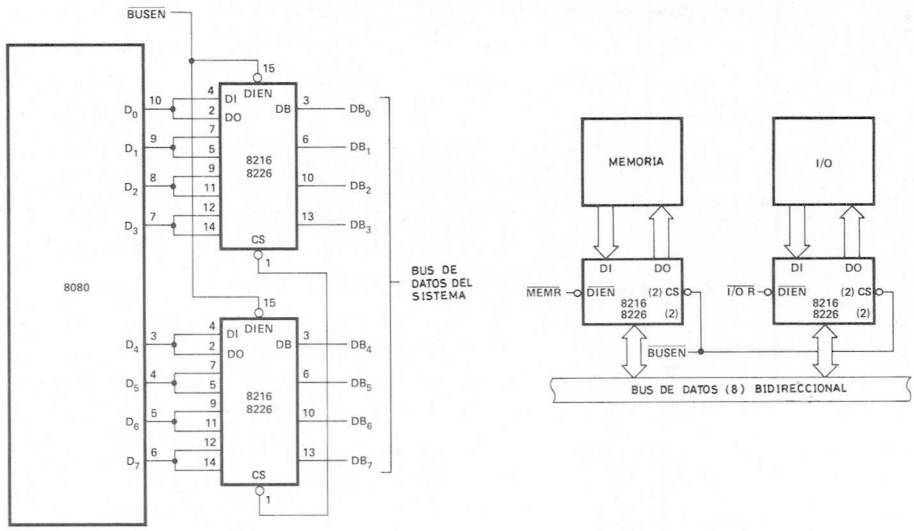


Fig. 7-46.–Conexión del 8216 a la CPU y a elementos de memoria y de I/O.

FAMILIA MICROCOMPUTADOR PARA EL 8085 (MCS-85)

La CPU 8085, como ya se ha explicado, es mucho más potente que la 8080 y, mientras que las funciones básicas del 8080 hay que realizarlas junto con 2 chips auxiliares, el 8085 no los necesita. En la figura 7-47 se muestra la equivalencia entre los 2 sistemas.

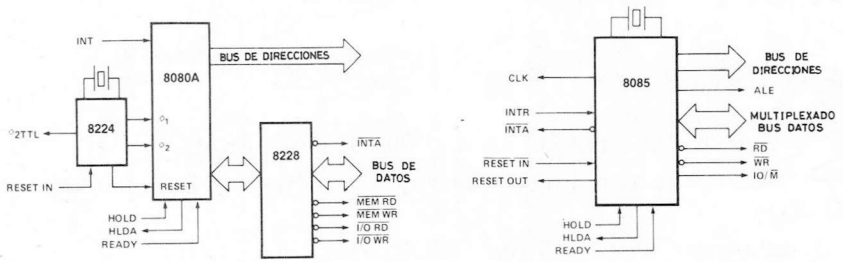


Fig. 7-47.–Las funciones básicas con la CPU 8080 requieren más elementos que la CPU 8085.

La familia MCS-85 dispone de elementos mucho más potentes que los estudiados para la familia MCS-80 y que se describen someramente.

El circuito 8155/8156 está formado por:

2048 bits organizados en 256 palabras de 8 bits cada una (RAM).
2 puertas programables de I/O, de 8 bits.
1 puerta programable de 6 bits de I/O.
1 Contador-Temporizador programable de 14 bit.

El esquema simplificado del 8155 así como sus registros internos se muestran en la figura 7-48.

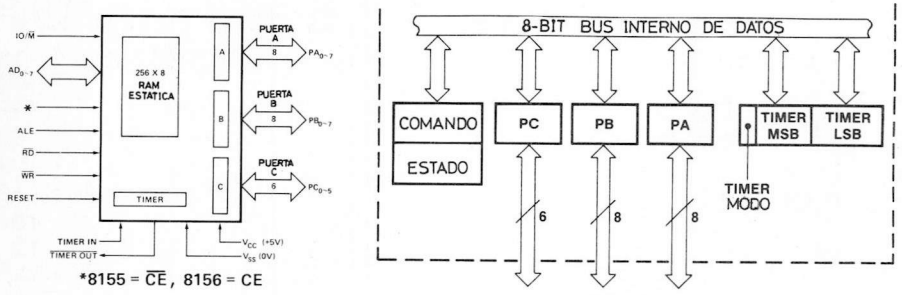


Fig. 7-48.–Estructura y registros internos del 8155/8156.

Otro circuito muy importante de la familia MCS-85 lo constituye el 8355/8755, que contiene:

16.384 bits ROM/EPROM distribuidos en 2048 palabras de 8 bits.
2 puertas de 8 bits para aplicaciones generales.

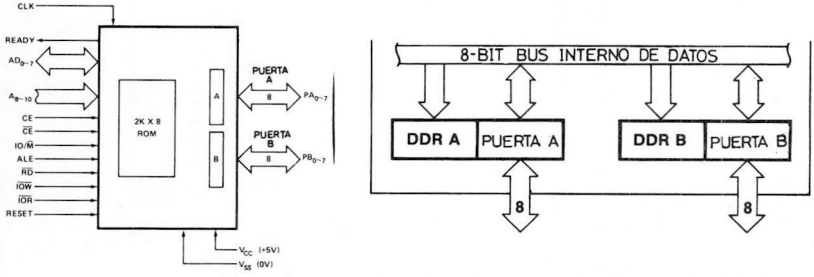


Fig. 7-49.–Estructura y registros internos del 8355.

Cada línea de I/O de las puertas es programable independientemente.

Pines compatibles con la ROM 8355 y la EPROM 8755 entre sí.

La figura 7-49 muestra la estructura y registros internos del 8355.

A base de la CPU 8085 y de los dos chips de la familia MCS-85, ya comentados, se puede implementar un sistema mínimo microcomputador como el que aparece en la figura 7-50.

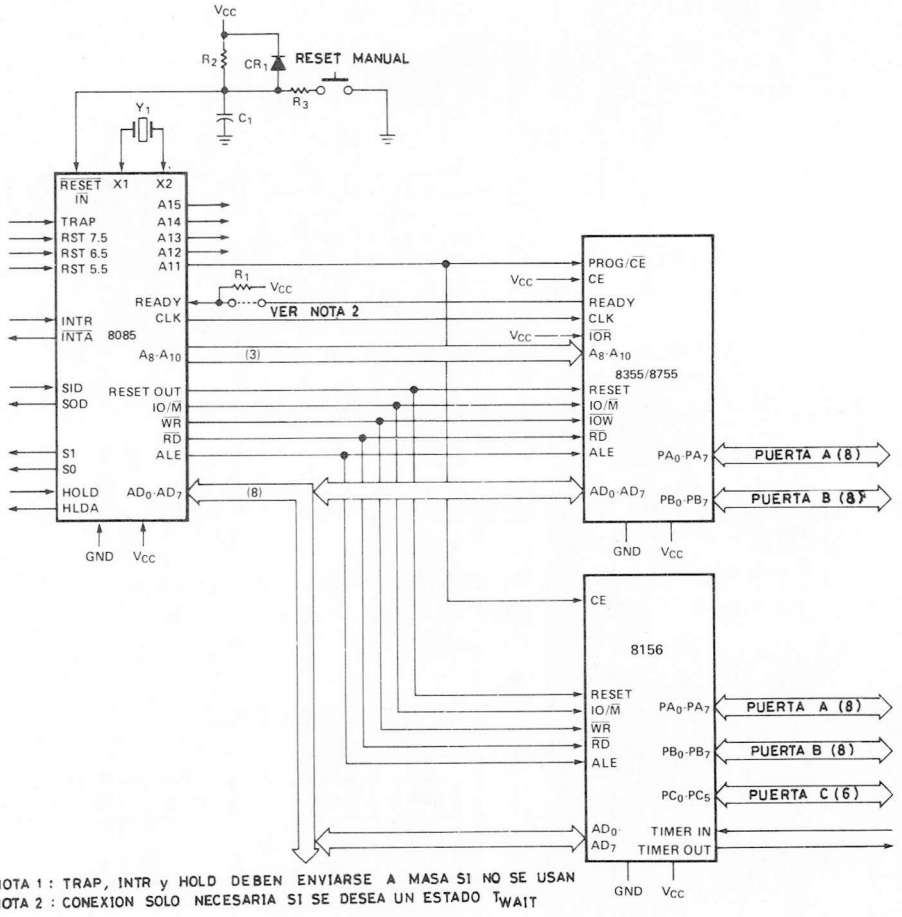


Fig. 7-50.–Sistema mínimo microcomputaodr con elementos de la familia MCS-85.

Por otra parte, este sistema mínimo es fácilmente ampliable, como se expone en la figura 7-51.

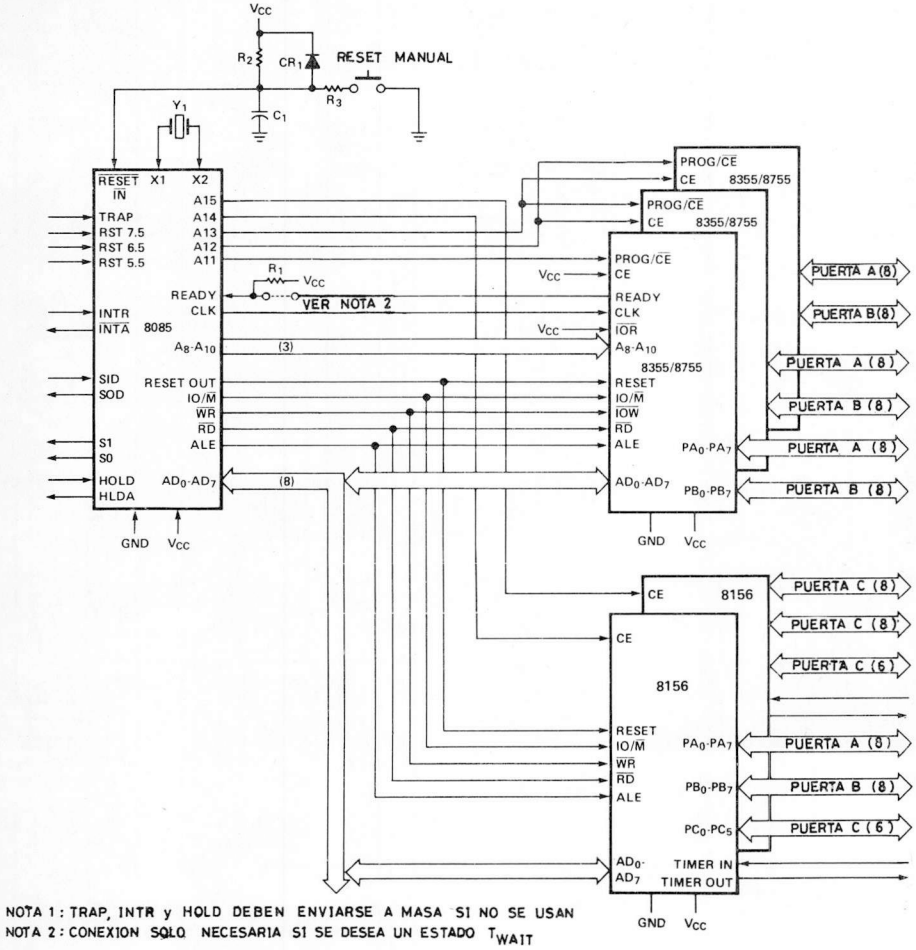


Fig. 7-51.–Expansión del sistema mínimo de la familia MCS-85.

INTERCONEXION DE LA CPU 8080 CON LA MEMORIA Y LOS DISPOSITIVOS DE ENTRADA Y SALIDA

La interconexión de la CPU con otros elementos se realiza sencillamente a través de las señales de control y los buses de datos y direccionamiento.

En la figura 7-52 se ofrece un sistema típico, basado en el microprocesador 8080, que puede utilizarse como base para cualquier desarrollo independientemente de su tamaño o complejidad.

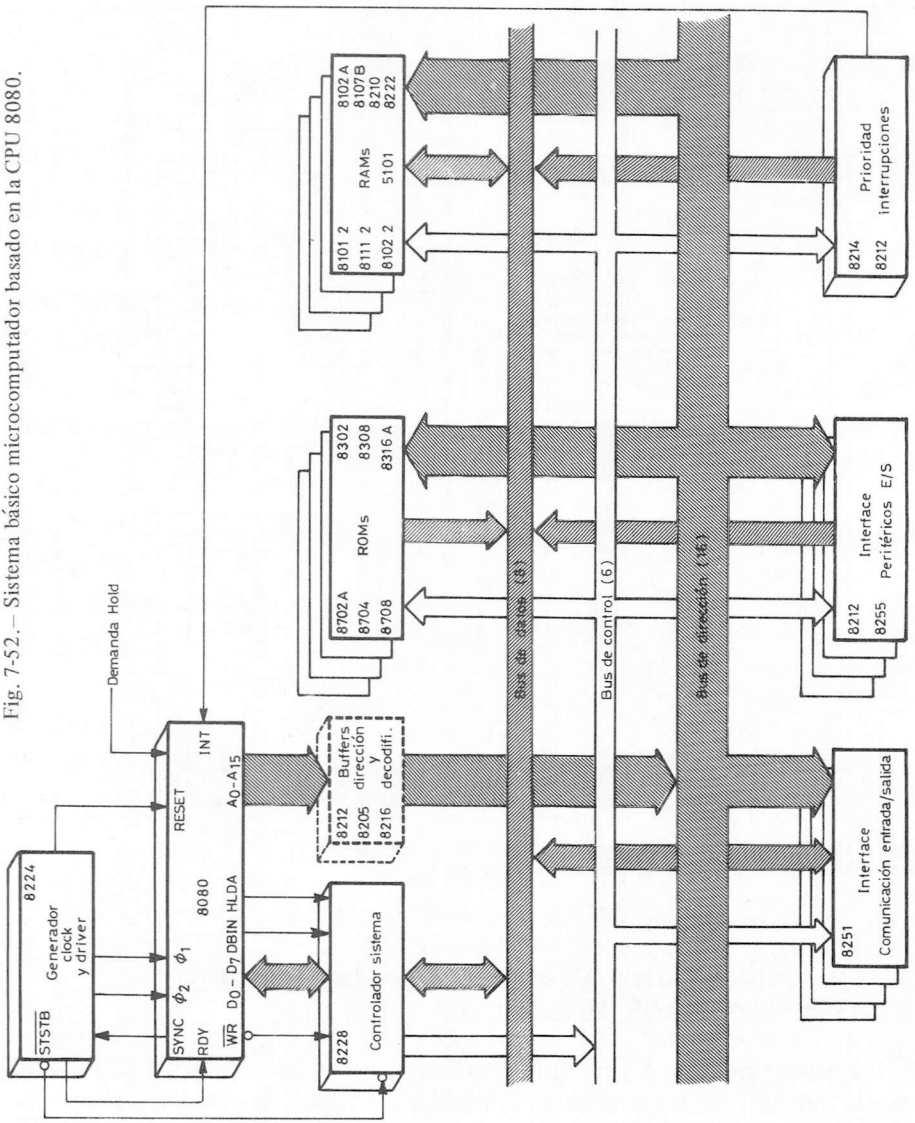


Fig. 7-52.– Sistema básico microcomputador basado en la CPU 8080.

Obsérvese en la figura 7-52 que la conexión con la memoria y los dispositivos de entrada/salida es exactamente la misma, dependiendo la operación a ejecutar de las señales de lectura/escritura del bus de control. Ello permite al 8080 configurar un sistema en el que la memoria y las entradas y salidas constituyen una sola área para pequeños sistemas con menos de 32 K bytes de memoria.

Conexión con memorias ROM

La conexión de la CPU a las memorias ROM es muy simple y se realiza directamente. Las líneas de salida de datos de la memoria se conectan al bus de datos bidireccional, las entradas de direccionamiento al bus de direcciones, siendo posible la decodificación de los bits de más peso de la dirección para ser utilizados como "selección de chip o circuito" y conectando la señal MEMR del bus de control a la "selección de chip" o al buffer de datos.

Básicamente, la CPU envía una dirección de memoria durante la primera parte del ciclo de búsqueda de la instrucción (T1 y T2). Esta información, colocada en el bus de dirección, selecciona una posición determinada en la ROM, colocándose unos instantes después (dependiendo del tiempo de acceso) los datos almacenados en la posición direccionada en las líneas de salida de datos. En este instante, (T3), el bus de datos de la CPU está en el modo de entrada, enviando la lógica de control una orden de lectura de memoria (MEMR), que coloca la información procedente de la ROM en el bus de datos de la CPU.

Conexión con memoria RAM

Para leer datos de la memoria RAM, la CPU realiza exactamente el mismo proceso que con la memoria ROM, siendo muy parecida aunque no idéntica la operación de escritura.

La CPU envía a la RAM una dirección de memoria durante la primera parte del ciclo de escritura en memoria (T1 y T2), siendo en el estado T3 cuando se sacan de la CPU los datos a escribir en la memoria, colocándose en el bus de control la señal MEMW. Esta señal se conecta a la entrada R/W de la RAM, dejando accesible la posición seleccionada para la carga de datos.

La figura 7-53 muestra un sistema típico de memoria, en el que se aprecia cómo se conectan a los buses los componentes de la memoria. Esta tiene un tamaño de 8 K bytes de memoria ROM, empleando para su

implementación circuitos Intel 8316 y 512 bytes de memoria RAM a base de chips 8111 de Intel. El interface básico con la estructura del bus es común para cualquier tamaño de memoria, debiéndose añadir únicamente para sistemas mayores más buffers (8216-8212) y decodificadores para generar las señales de "selección de chip".

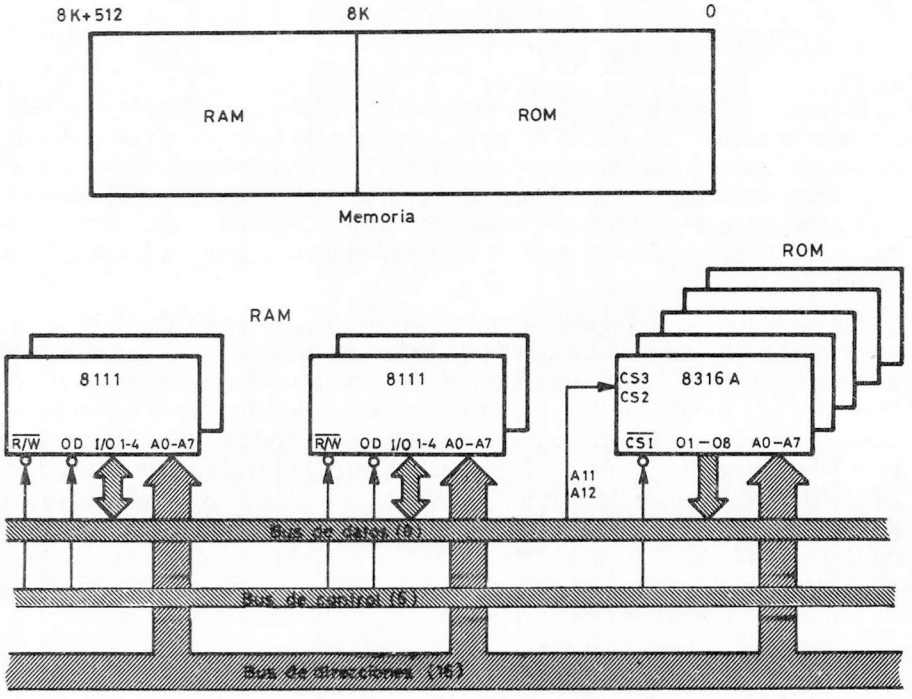


Fig. 7-53.–Interface típico de memoria.

Las memorias empleadas en la figura 7-53 tienen un tiempo de acceso de unos 850 ns, siendo dispositivos algo lentos, pero económicos y fáciles de conectar al 8080 sin mermar sus posibilidades. Cuando la CPU opera con una base de tiempo de 500 ns, se requiere una memoria con un tiempo de acceso de 450-550 ns (como los modelos de Intel: 8308, 8102 A, 8107 A, etc.), pudiendo permanecer constantemente a 1 la línea READY de la CPU, puesto que no se necesitan estados de espera (WAIT). No obstante, si se utilizan memorias más lentas, como las del ejemplo, que tienen un tiempo de acceso mayor que el mínimo requerido por la entrada READY, la CPU 8080 coloca un estado adicional de espera, que equivale

a uno o más períodos de reloj, para compensar el retardo del componente. Ello supone una ejecución más lenta de la instrucción, obteniéndose con un sólo período de espera adicional un ciclo de instrucción de 2,5 microsegundos.

Interconexiones de entrada y salida

Los dispositivos I/O tales como teclados, lectoras de cinta de papel floppy disk, impresoras, etc., se usan para introducir u obtener información de la CPU y para mostrar o almacenar los resultados consecuencia de un proceso.

Una de las características más importantes del sistema 8080 es precisamente la flexibilidad y potencia de su estructura de entrada/salida y los componentes para soportarla. Según el sistema, existen varias formas de estructurar las entradas/salidas, el fin que se persigue y la máxima eficiencia con el menor número de componentes.

Para comprender más fácilmente la estructura de las entradas/salidas, pueden considerarse como un conjunto de bytes de memoria, en los que se puede leer o escribir. La CPU 8080 tiene instrucciones específicas para realizar transferencias con las entradas/salidas (IN, OUT). Estas instrucciones aislan la memoria de las entradas/salidas, de tal forma que la zona

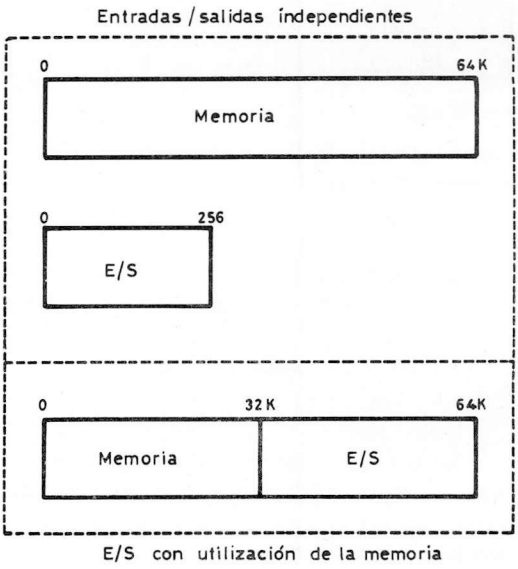


Fig. 7-54.–Mapa de la memoria en el uso de Entrada/Salida.

de direcciones de memoria no se ve afectada por la estructura de I/O, realizándose la transmisión de una forma tan simple como la de enviar a o desde el Acumulador a una puerta direccionada.

Otra forma es la de considerar la estructura de I/O como una parte de memoria. Este método permite utilizar un nuevo grupo de instrucciones muy potente para la manipulación de entradas y salidas (figura 7-54).

Las 2 formas de tratar a las entradas y salidas se comentan a continuación.

ENTRADAS/SALIDAS INDEPENDIENTES: En este tipo de estructura de I/O el área de direcciones de memoria y el de direcciones de entrada/salida están aisladas entre sí, y la transferencia se realiza siempre a partir del Acumulador. Las operaciones en esta estructura son muy simples, pues únicamente se dispone de las instrucciones IN (entrada de datos al Acumulador desde una puerta de entrada) y OUT (salida de datos desde el Acumulador a una puerta de salida). Al estar completamente independientes las direcciones de los dispositivos de I/O, se puede direccionar la máxima capacidad de memoria, es decir, 64 K, sin afectar para nada al direccionamiento de los periféricos (figura 7-55).

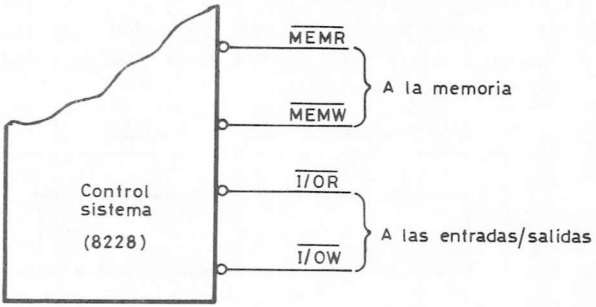


Fig. 7-55.–Entradas y Salidas independientes.

ENTRADAS/SALIDAS CON UTILIZACION DE MEMORIA: Asignando una zona de memoria a las I/O, se puede desarrollar una potente estructura, que permite utilizar en las transferencias de I/O las mismas instrucciones que las que se usan para manipular las posiciones de memoria. Esto es, pues crearse un nuevo grupo de instrucciones para gobernar las I/O. A este sistema se le llama "mapa de memoria".

En la figura 7-56 se combinan las señales de control MEMR y MEMW con el bit de más peso de la dirección de memoria (A15), generándose unas nuevas señales de control, que pueden conectarse, exactamente igual que en el caso de I/O, independientes, ya que las características del bus permanecen invariables.

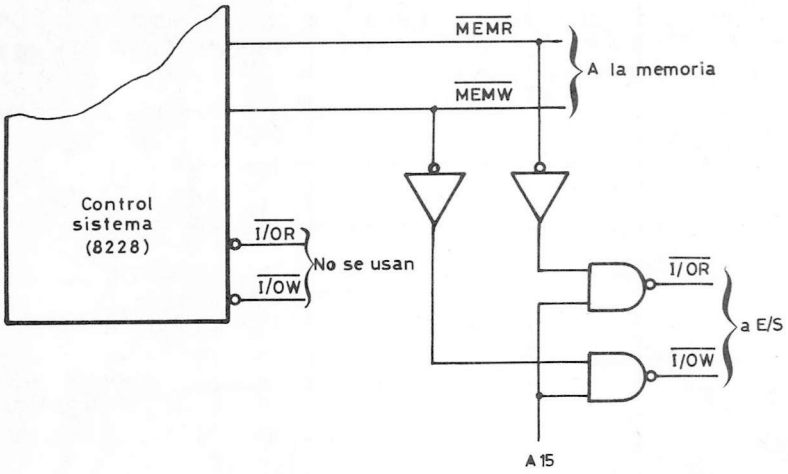


Fig. 7-56.–Entrada/salida con utilización de memoria.

Asignando al bit A15 la condición de determinación de entradas/salidas, debe observarse un sencillo método de discriminación:

Si A15 es cero, la memoria es activa.
Si A15 es uno, son activas las entradas/salidas.

Es aconsejable utilizar el A15 porque, al ser el más significativo del direccionamiento, define más claramente las dos zonas de memoria, quedando 32 K para empleo en cada zona.

Con esta técnica los dispositivos de I/O se consideran puertas direccionadas y se pueden utilizar todas las operaciones que actúan sobre la memoria, como MOV r,M; MOV M,r; MOVI M; LDA; STA; LHLD; SHLD; ADD M, etc.

Empleando este sistema para las I/O, aunque se limita el área de la memoria, se obtiene un aumento significativo en la velocidad del proceso global del sistema, así como una reducción en la zona de memoria requerida por el programa.

La figura 7-60 muestra un sistema típico de I/O, utilizando diversos dispositivos (8212, 8521 y 8255). Este interface se podía utilizar con diversos periféricos, como un CRT, un teclado, un display y un interface de transmisión en serie. Otra aplicación muy distinta podría ser, en un control de proceso, el interface de sensores, relés o control de motores.

El circuito 8251 proporciona el interface para un sistema de transmisión de datos en serie, en el que la emisión y la recepción se pueden realizar por líneas telefónicas (figura 7-57).

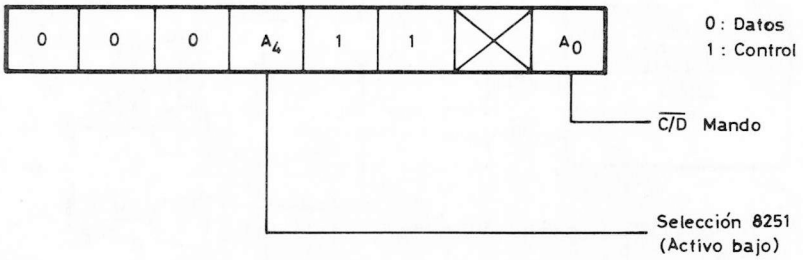


Fig. 7-57.–Selección del 8251.

Cada uno de los dos circuitos 8255 proporciona 24 bits de datos de entrada o salida programables para control de teclados, sensores, lectoras de cinta, etc., interconectables con el sistema (figura 7-58).

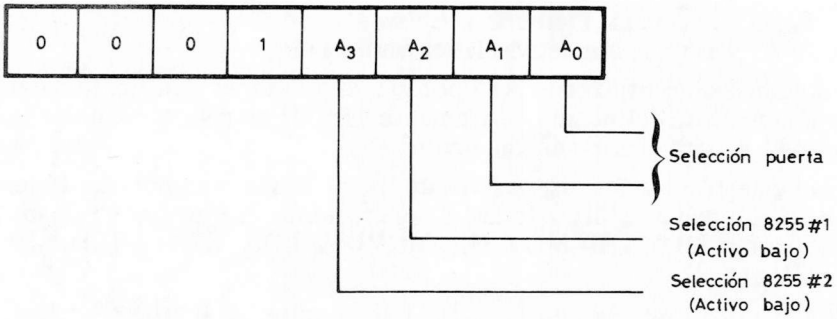


Fig. 7-58.–Formato para la selección del 8255.

Los 3 circuitos 8212 pueden utilizarse para líneas de consumo elevado o indicadores luminosos tipo LED, debido a su gran capacidad de carga a la salida (15 mA) (figura 7-59).

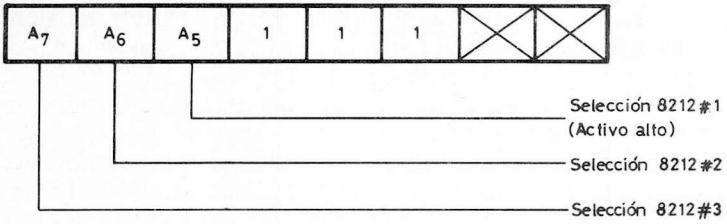


Fig. 7-59.–Formato de seleccionamiento de los circuitos 8212.

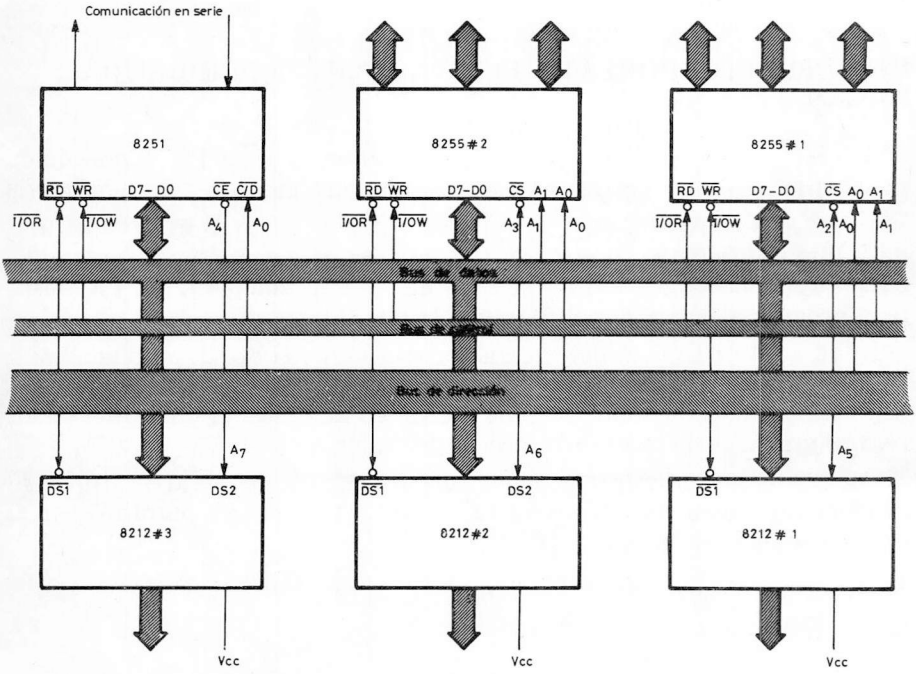


Fig. 7-60.–Interface típico de Entradas/Salidas.

MICROMPUTADORES INTEL EN UN SOLO CHIP: 8048 Y 8748

Los chips 8048 y 8748 de Intel contienen todo un sistema microcomputador, comprendiendo: CPU, Memoria de programa de 1 K bytes, Memoria de datos de 64 bytes, generador de clock, Contador/Tempori-

zador y 27 líneas para entradas y salidas. La única diferencia entre los 2 chips estriba en que el 8048 tiene la memoria de programa en ROM, mientras que el 8748 la tiene en PROM, teniendo ambos sus pines compatibles e intercambiables entre sí. Esta diferencia determina la utilización del 8748 para prototipos o pequeñas series y al 8048 para grandes producciones. Para la posible ampliación de estos chips microcomputadores se recomienda el uso del 8755 (2 K X 8 EPROM + 15 líneas I/O) o del 8355 (2 K X 8 ROM + 16 I/O) o del ya estudiado, 8155. Dada la importancia de esta familia de chips, más adelante se hace una exposición detallada de los mismos.

MICROPROCESADORES INTEL DE APLICACION INDUSTRIAL: ID 8049 E ID 8039

Se trata, en realidad, de conjuntos alrededor de una CPU, especialmente orientados a su aplicación en los diversos campos de la industria.

El ID 8039 dispone de 128 bytes de RAM, un contador/temporizador de 8 bits y 27 líneas de entrada y salida. Se ha diseñado para su empleo en el desarrollo de prototipos en conjunción con una memoria PROM para el programa, accesible y programable por el usuario.

El ID 8049 tiene similares características que el 8039, pero además se incluye en el chip 2 K bytes de memoria ROM, lo que hace aconsejable utilizarlo en producciones de cierta cantidad de chips, al tener que ser necesario grabar la memoria en fábrica.

Ambos chips pueden trabajar a una frecuencia de 11 MHz, todas sus instrucciones constan a lo sumo de 2 octetos (2 bytes) y permite el tratamiento aritmético binario y BCD.

APENDICE

PERIFERICO PROGRAMABLE PPI 2655

INTRODUCCION

Al igual que otros fabricantes de microprocesadores, SIGNETICS ofrece un conjunto de chips, que facilitan la implementación de sistemas basados en su microprocesador 2650.

El 2655 se ha concebido para ser usado como periférico del 2650 y de otros microprocesadores. Dispone de 24 terminales de entrada y salida, divididos en 3 grupos de 8, denominados registro A, registro B y registro C.

Puede operar de 7 formas diferentes:

- E/S estáticas.
- Selección de entrada.
- Selección de salida.
- E/S bidireccionales seleccionadas.
- Entrada serie.
- Salida serie.
- Contador de sucesos y tiempos.

El 2655 se puede emplear en diversas aplicaciones, consiguiéndose una reducción del número de circuitos integrados y simplificando en gran manera el hardware del sistema, según la Revista Miniwatt.

DESCRIPCION DEL CIRCUITO

La figura 7-61 muestra el diagrama bloques del Periférico Programable (PPI) 2655.

Todas las transferencias de información y control de programación entre el PPI y el microprocesador se efectúan a través del amplificador de la barra de datos, el cual enlaza las barras de datos del microprocesador y el periférico.

El acceso al PPI se controla mediante la entrada de inhibición ($\overline{CE}$), mientras que el acceso a los distintos registros se selecciona mediante los bits de dirección A_0 y A_1. La señal de Lectura/Escritura ($\overline{R}/W$) selecciona el sentido de circulación de la información entre el PPI y el microprocesador.

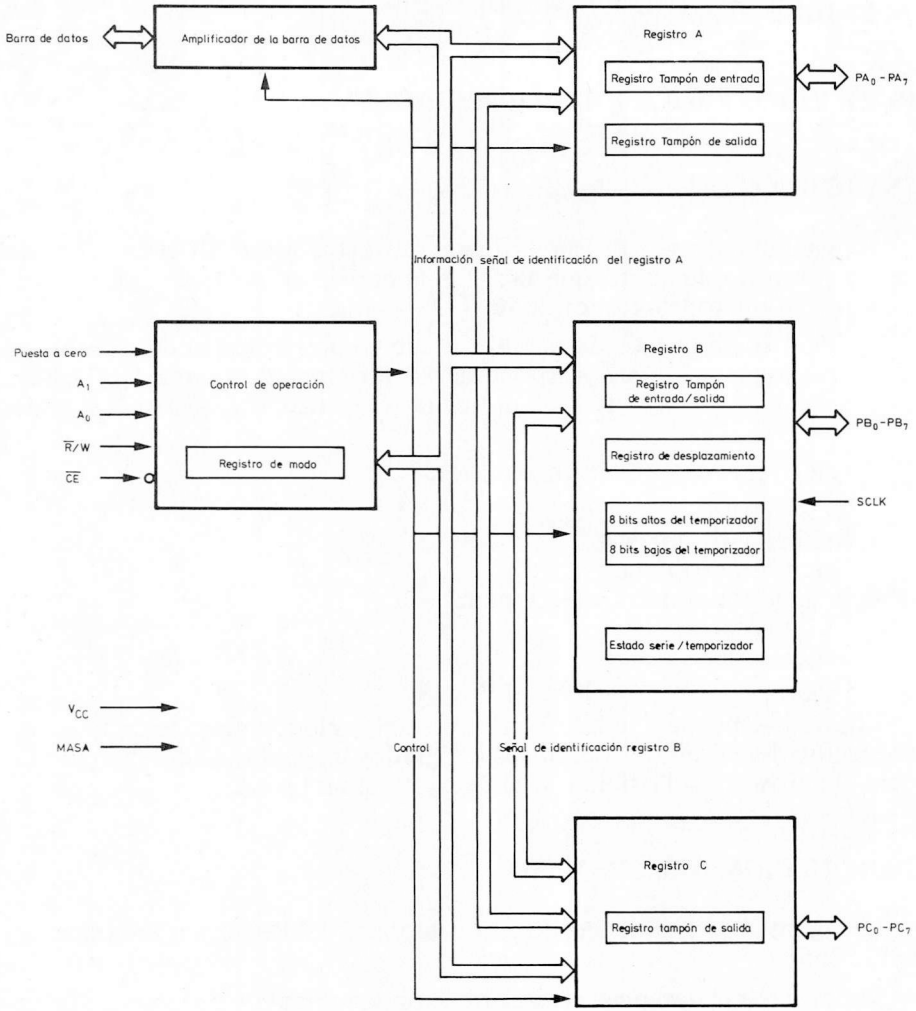


Fig. 7-61.—Diagrama por bloques del periférico programable 2655.

Los distintos modos de operación del PPI se seleccionan mediante la palabra de modo, escrita en el registro de modo por el microprocesador.

En la palabra de modo, tres bits seleccionan el modo de operación del registro A, que puede ser:

- E/S estáticas,
- entrada seleccionada,
- salida seleccionada,
- E/S bidireccionales seleccionadas.

Otros tres bits seleccionan el modo de operación del registro B, que puede ser:

- E/S estáticas,
- entrada seleccionada,
- salida seleccionada,
- entrada/temporizador en serie,
- salida/temporizador en serie.

Obsérvese que los modos serie y temporizador operan simultáneamente.

Para el registro C no es preciso indicar modo de operación, dado que éste puede operar solamente en el modo E/S estáticas. Sin embargo, el número de bits del registro C, que puede utilizarse para la operación E/S estática, es limitado cuando el registro A y/o el B precisan terminales del registro C. Esto sucede mientras se selecciona el modo de operación que es cuando se precisa un número elevado de señales de control. Esto se demuestra en la figura: los registro A y B pueden intercambiar información con el microprocesador y los periféricos a través del registro C.

Así pues, en la programación de los registros A y B pueden emplearse terminales del registro C; los bits correspondientes del registro C pueden ser leídos por el microprocesador para conocer el estado de la palabra de modo de los registros A y/o B. Para conocer el estado del registro B, en el modo de operación serie/temporizador, se ha previsto un registro especial al que puede acceder el microprocesador para su lectura.

Cualquier bit de un registro específico puede ponerse a los niveles lógicos cero o uno mediante el bit set/reset. Debe procurarse evitar la pérdida de los datos introducidos, o los bits de estado o los de control debido a la posibilidad de modificación de éstos con la orden set/reset. Dos de los ocho bits de la palabra de selección direccionan a uno de los tres registros, o al registro serie/temporizador A. Tres bits direccionan el bit deseado del registro seleccionado y un bit indica si es una orden de puesta a cero o de puesta a uno.

MODOS DE OPERACION

E/S estáticas

La figura 7-61a muestra la estructura básica de un terminal de E/S estáticas. El registro tampón puede verse como si tuviera una salida de drenaje abierta con un resistor conectado a positivo. Los datos escritos en el registro tampón por el microprocesador están presentes en los terminales E/S; el microprocesador puede leer también la información almacenada. Cuando en el terminal de las E/S aparece un uno lógico, la señal puede ser controlada por una entrada a nivel cero de un componente exterior. Del mismo modo, una entrada en nivel uno en un dispositivo exterior puede ser controlada por un bit de salida que esté en nivel cero: por consiguiente, el registro tampón debe cargarse con uno lógico antes de las operaciones de entrada. La circuitería interna del 2655 regula automáticamente los niveles de impedancia de las entradas/salidas para cada operación, sea de entrada o de salida, y proporciona también protección contra cortocircuitos del terminal a masa o a V_{CC}.

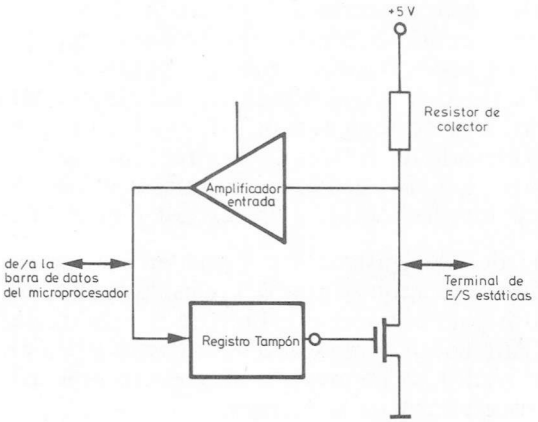


Fig. 7-61 a.–Estructura básica del terminal de E/S estáticas.

Cada uno de los registros A, B y C puede operar en el modo E/S estáticas, y es posible cualquier combinación de los terminales de entrada y salida. Cada registro puede ser leído o escrito por el microprocesador octeto a octeto, mientras que cada bit puede ponerse a cero o a uno mediante la orden set/reset. Cuando se utiliza un registro para una combinación de entradas y salidas, los bits correspondientes a los terminales de

entrada deben conectarse formando una función "0" con un nivel lógico uno mediante una máscara por software al escribir una palabra de 8 bits en el registro. Del mismo modo, una máscara por software puede separar la información de entrada y salida al leerla del registro.

Entrada y salida seleccionadas

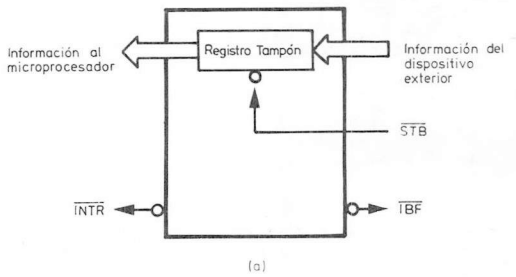
La operación de selección precisa el intercambio de señales de control entre el microprocesador y el PPI y entre el PPI y los periféricos. Este intercambio de señales es denominado de identificación.

La figura 7-61b muestra las señales requeridas por las identificaciones de la entrada de datos y la temporización.

La figura 7-62 muestra las señales precisas para la salida de datos.

$\overline{STB}$: Selección; permite que el registro tampón de entrada acepte o retenga la información del dispositivo exterior.

$\overline{IBF}$: Llenado del Registro de Entrada; informa al dispositivo exterior que el registro tampón de entrada ha aceptado su información pero que el microprocesador no la ha leído todavía.



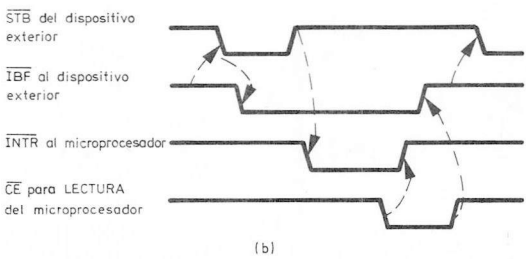


Fig. 7-61 b.– Señales de entrada seleccionada (a) y sincronizador de la señal de identificación (b).

$\overline{\text{INTR}}$: Petición de interrupción; *para una señal de identificación de entrada* informa al microprocesador que se ha escrito nueva información en el registro tampón de entrada; *para una señal de identificación de salida* informa al microprocesador que el dispositivo exterior ha aceptado la información contenida en el registro tampón de salida.

$\overline{\text{OBE}}$: Vaciado del Registro Tampón de Salida; informa al dispositivo exterior que el microordenador no ha escrito todavía nueva información de salida en el circuito de retención de salida.

$\overline{\text{ACK}}$: Acuse de Recepción: indica que los dispositivos exteriores han aceptado la información del registro tampón de PPI.

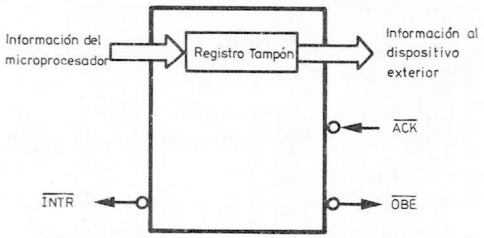
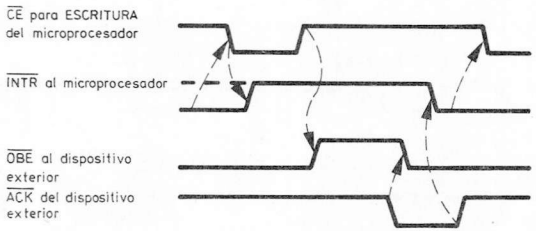


Fig. 7-62.—Señales de salida seleccionada en la parte superior y sincronización de la señal de identificación en la parte inferior.



Debido a que el número de terminales es insuficiente para asignar un terminal a cada señal de identificación, deben sacrificarse algunos de los terminales de E/S estáticas del registro C. Esto significa que cuando el registro A opera en los modos de selección, tres de los terminales de E/S estáticas del registro C son utilizados por las señales de identificación. Del mismo modo, los modos de selección del registro B precisan tres terminales del registro C, lo que significa que si el registro A y/o el registro

B operan en el modo de selección, el registro C tiene solamente 2 o 5 terminales utilizables por las E/S estáticas. La figura 7-63 muestra las señales asignadas a los terminales del registro C para entradas/salidas seleccionadas en los registros A y B.

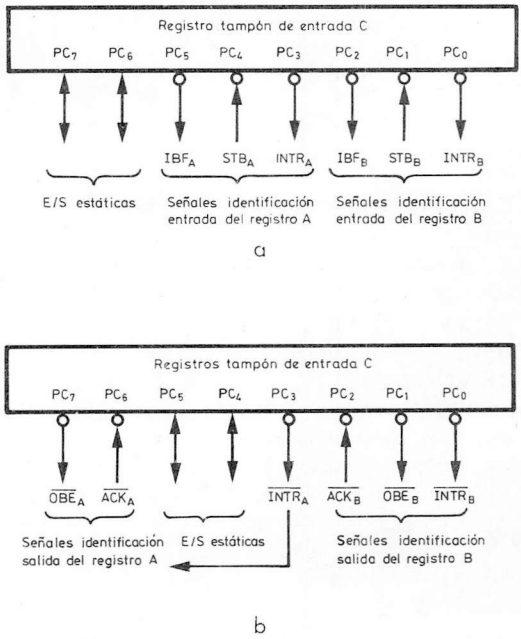


Fig. 7-63.–Asignación de señal al registro C para entrada (a) y salida (b), seleccionadas en los registros A y B.

El microordenador puede leer el estado de la señal de identificación mediante la lectura del registro C. La figura 7-64 muestra el significado de cada uno de los bits de dicho registro. Los bits INTE (inhibición de interrupción) pueden activarse o ponerse a cero para inhibir o desinhibir las señales de INTR (petición de interrupción) a los terminales del registro C.

Todos los terminales del PPI 2655, utilizados para las señales de petición de interrupción INTR, son de drenador abierto, de forma que pueden combinarse entre sí para proporcionar una sola señal de petición de interrupción. Entonces, el microordenador puede identificar qué señal es la que ha producido la petición de interrupción, comprobando el estado de la señal de identificación.

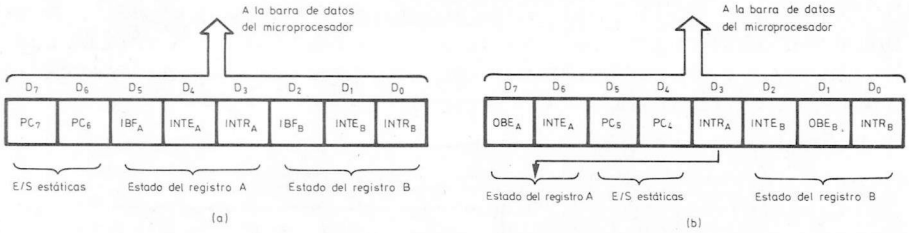


Fig. 7- 64.–Información de estado del registro C al microprocesador.

E/S bidireccionales seleccionadas

Solamente el registro A puede operar en este modo (figura 7-65). Es una combinación de los modos de entrada y salida seleccionadas, con la petición de interrupción para entrada y salida como una sola señal. Así pues, este modo de operación precisa cinco terminales del registro C para sus señales de identificación. Si el registro B está operando en modo de entrada y salida, todos los terminales del registro C se utilizan para las señales de identificación.

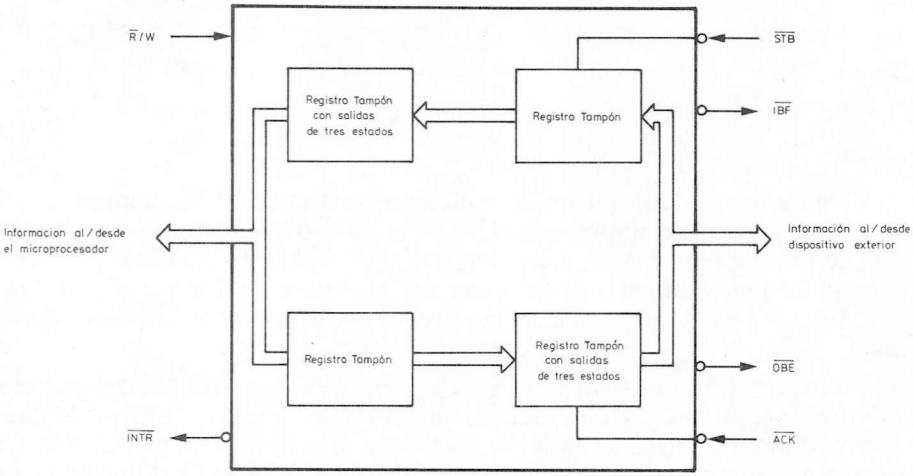


Fig. 7-65.–Señales en el modo de E/S bidireccionales seleccionadas.

La figura 7-66 muestra la asignación de la señal. El microordenador puede leer el estado de la identificación mediante la lectura del registro C (figura 7-66). Obsérvese que ahora hay dos señales de estado de inhibición de interrupción, $INTE_{IN}$ e $INTE_{OUT}$. Estas pueden activarse o ponerse a cero con las órdenes de control del registro C.

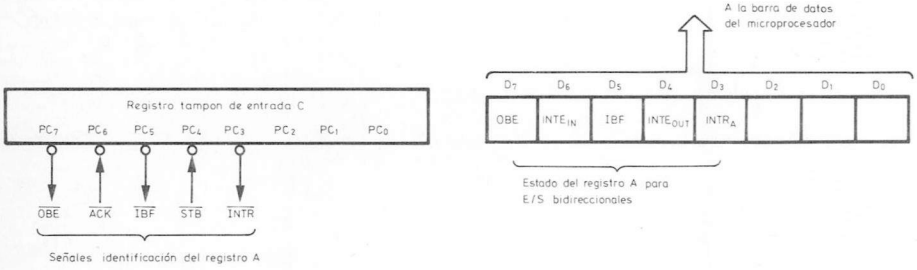


Fig. 7-66.—A la izquierda, asignación de las señales de identificación de E/S bidireccionales del registro A al registro C. A la derecha, información de estado para el modo de E/S bidireccionales del registro A en el registro C.

Modo serie/temporizador

Solamente el registro B puede programarse en el modo serie/temporizador. El modo temporizador opera siempre simultáneamente con el modo de entrada serie o con el modo de salida serie. La figura 7-67 muestra la función de los terminales del registro B en el modo serie/temporizador.

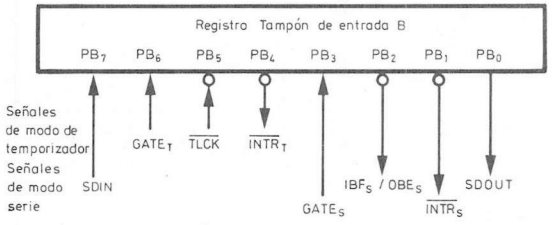


Fig. 7-67.—Asignación del terminal del registro B en el modo serie/temporizador.

Modos de entrada y salida serie

En la figura 7-68 se muestran los modos serie de entrada y salida.

En el modo de entrada serie, la información en serie se convierte en palabras de información en paralelo y se envía al microprocesador. El

flujo de datos en serie de muestrea en el flanco de subida del SCLK (Reloj Serie), mientras que $GATE_S$ debe suministrar el conjunto de la infomación, con lo cual el PPI conocerá qué bits en serie son válidos. Cada grupo de ocho bits válidos de información recibidos se colocarán en el registro tampón de entrada del registro B para que el microprocesador los lea. Las señales de identificación son similares a las de los modos de selección: $INTR_S$ (Petición de Interrupción Serie) informa al microproce-

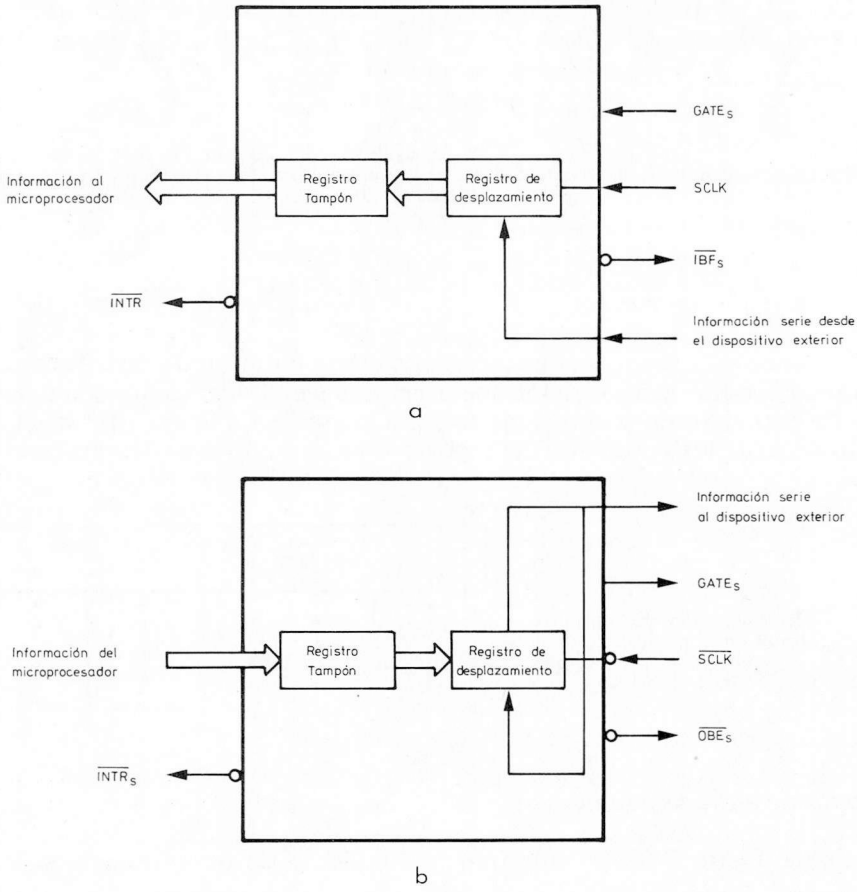


Fig. 7-68.–Principio de los modos de entrada (a) y salida (b) en serie.

sador que se ha recibido y almacenado en el registro tampón de entrada del registro B una palabra de información; $\overline{IBF}_S$ (Llenado Serie del Registro de Entrada) informa al dispositivo exterior que esta información convertida es utilizable por el microprocesador, pero que todavía no ha sido leída.

Si $GATE_S$ se desactiva antes de recibir una palabra completa de 8 bits en serie, el contenido del registro de desplazamiento en aquel momento se transfiere al registro B. Puede usarse para convertir palabras en serie de menos de 8 bits, en palabras en paralelo.

El modo serie de salida se utiliza para la conversión de octetos de información en serie. La información colocada en el registro B por el microprocesador, se transfiere al registro de desplazamiento y se serializa a la velocidad de régimen de la señal de entrada $\overline{SCLK}$.

La señal $GATE_S$ tiene dos funciones. Como salida, dirige las transferencias válidas de información. Toma el valor uno cuando se serializa un "nuevo" octeto de datos y toma el valor cero cuando se vuelve a transmitir un octeto anterior de datos, debido a un fallo de la CPU para cargar a tiempo el registro B. Si se fuerza la salida al estado cero cuando está en nivel lógico uno, las siguientes acciones dependen del estado del registro tampón en el momento en que la salida pasa al estado cero.

a) Si la CPU ha cargado un nuevo octeto de información en el registro tampón, el octeto se transfiere al registro de desplazamiento y se serializa cuando se desinhibe la señal $GATE_S$; $GATE_S$ pasará a nivel alto, indicando así la transmisión de esta nueva información.

b) Si se carga el nuevo octeto de información después del flanco de bajada de $GATE_S$, la información transmitida al desinhibirse $GATE_S$, será el resto del octeto anterior. $GATE_S$ permanecerá en nivel bajo durante este tiempo. Según esto, la nueva información se obtendrá con $GATE_S$ a nivel alto, para indicar que la transmisión de la información es válida.

Esta técnica puede utilizarse para controlar el número de bits que se serializa en cada octeto. La señal $GATE_S$ debe permanecer en nivel lógico cero durante un ciclo de $\overline{SCLK}$, como mínimo.

El modo serie utiliza cinco terminales del registro B: uno para la Salida de Información en Serie (SDOUT), uno para la Entrada de Información en Serie (SDIN) y tres para las señales de identificación $\overline{INTR}_S$, $GATE_S$ y $\overline{IBF}_E/\overline{OBE}_S$. El Reloj Serie utiliza un terminal independiente.

Para la modalidad serie/temporizador se ha previsto un registro especial de estado al que puede acceder el microprocesador para su lectura.

Modo de temporizador

El modo de temporizador da al PPI 2655 la posibilidad de efectuar medidas de amplitud de impulsos, recuento de sucesos, recuento de retardos y recuento de fin de sucesos.

Medida de amplitud de impulsos

El tiempo que transcurre entre los flancos de subida y bajada de la señal $GATE_T$ puede medirse contando el número de impulsos de sincronización de una fuente de frecuencia constante, alimentando la entrada del TCLK, tal como se muestra en la figura 7-69.

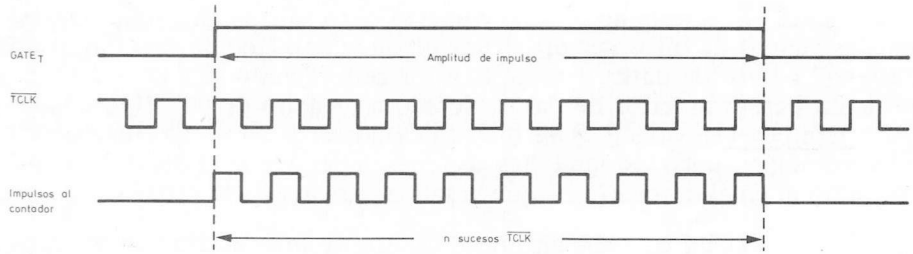


Fig. 7-69.–Medida de la amplitud de impulso en el modo de temporización.

Recuento de sucesos

Puede contarse el número de sucesos que ocurren durante el tiempo en que la señal $GATE_T$ es activa, cuando cada suceso proporciona un impulso a la entrada del TCLK, como se muestra en la figura 7-70.

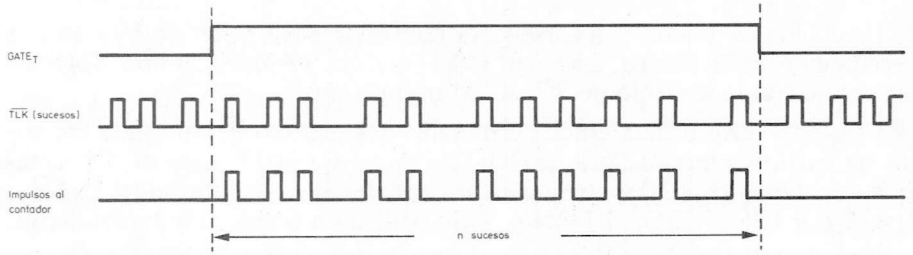


Fig. 7-70.–Recuento de sucesos en el modo de temporización.

Recuento de retardo

El microprocesador carga el contador con un valor inicial, y descuenta los impulsos TCLK (cuando la señal $GATE_T$ es activa), hasta que el contador llega a cero, en cuyo momento se genera un $INTR_T$ (Petición-Temporización) de interrupción para interrumpir al microprocesador. Durante el tiempo que transcurre entre la carga del contador y la petición de interrupción, el microprocesador puede ejecutar otros programas.

Recuento de sucesos de terminal

Es un recuento de sucesos hasta que se ha producido un número determinado de éstos. Se lleva a cabo cargando el contador con un número, después de lo cual irá disminuyendo éste a cada impulso de TCLK (sucesos), hasta que el contador llegue a cero. Esto genera una señal $INTR_T$, después de la cual el microprocesador puede actuar.

Estas funciones de sincronización pueden ejecutarse mediante la parte de temporización del registro B. Se utiliza un contador de 16 bits que disminuye su contenido a la velocidad del Reloj (TCLK) si la señal $GATE_T$ está en nivel activo, y genera una Petición de Interrupción ($INTR_T$) cuando llega a cero. El contador debe ser inicializado por el microordenador antes de iniciarse el contaje. Se inicializa con dos octetos, uno para los 8 bits superiores y uno para los 8 bits inferiores, a través de la barra de datos.

El recuento puede terminarse programando de nuevo el registro B, al mismo modo o a otro diferente, que también pone a cero la señal $INTR_T$.

El modo de temporizador utiliza tres terminales del registro B: uno para TCLK, otro para $GATE_T$ y uno tercero para $INTR_T$.

Registro del estado serie/temporizador

Los bits del registro de estado serie/temporizador pueden dividirse en cuatro grupos, como muestra la figura 7-71.

D_0 Control de acceso a modelo (MAC)

Los modos serie y de temporizador comparten el registro B y el registro de estado serie/temporizador. El acceso a cada uno de ellos lo controla el bit MAC. Este bit controla también cuál de los modos debe programarse y pone también a cero los bits apropiados de estado en el registro de estado serie/temporizador.

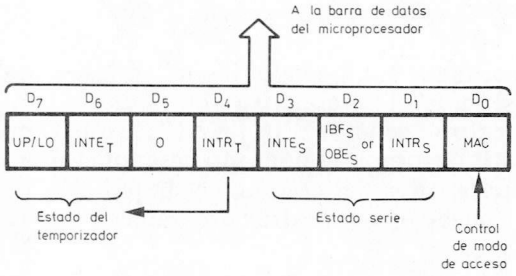


Fig. 7-71.–Información en el registro de estado serie/temporización.

$D_{1,2,3}$ Bits de estado E/S serie

Obsérvese que D_3 contiene el bit $INTE_S$ (Inhibición de Interrupción serie), que inhibe/desinhibe la señal $INTR_S$ en el temrinal PB_1. El bit $INTE_S$ puede ponerse al nivel lógico 0 o 1 con el bit de control de set/reset.

$D_{4,6,7}$ Bits del estado del temporizador

Obsérvese que D_6 contiene el $INTE_T$ (Inhibición de Interrupción-temporizador) que inhibe/desinhibe la señal $INTR_T$ en el terminal PB_4. D_7 selecciona a cuál de las partes del temporizador (superior o inferior) se accede, durante la lectura/escritura del/al temporizador. D_7 cambia de estado cada vez que se accede al temporizador. D_6 y D_7 pueden ponerse al nivel lógico 0 o 1 con el bit de control set/reset.

D_5 Este bit es siempre cero.

La figura 7-72 muestra un ejemplo en el que el PPI 2655 se programa para que tenga un acceso (registro A) E/S de 8 bits, un acceso de salida serie y un temporizador (registro B) y 3 terminales E/S estáticas, que se dejan en el registro C.

Obsérvese que todos los terminales del 2655, que se programan para INTR tienen una configuración de drenador abierto, de forma que pueden formar una función lógica "0" con un nivel uno, con un resistor conectado a positivo.

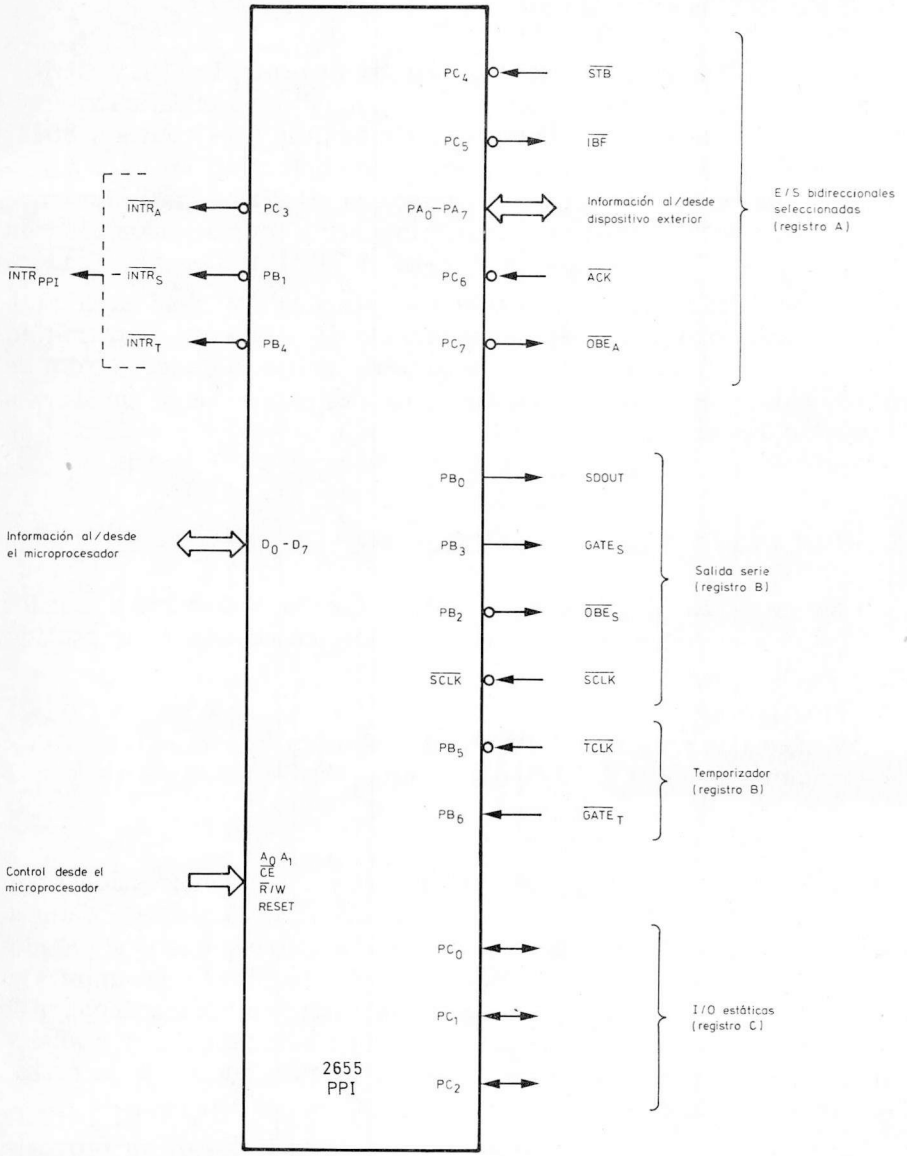


Fig. 7-72.–Ejemplo de programación de modo-múltiple del PPI 2655.

MICROORDENADOR 8048

En virtud del intercambio tecnológico suscrito entre INTEL y SIGNETICS, esta última dispuso de la posibilidad de fabricación del microordenador 8048 y de los componentes de la familia, tales como el 8021, 8022, 8035 y 8041. Aunque anteriormente se han mencionado las características fundamentales del 8048, se hace en este tema una extensa exposición de sus características, funcionamiento y programación ofrecida por COPRESA, distribuidor en España de PHILIPS y SIGNETICS.

La familia del microordenador 8048, denominada MCS-48 ha sido posible gracias a los últimos avances en tecnología NMOS que ha permitido fabricar en una única pastilla de silicio, un auténtico microordenador de un solo chip, que contiene todas las funciones requeridas en un sistema de proceso digital.

INTRODUCCION A LA FAMILIA MCS-48

El circuito más significativo de toda la familia es el microordenador 8048, el cual contiene las siguientes funciones en un único encapsulado de 40 patillas:

- CPU de 8 bits;
- memoria de programa ROM de 1 K X 8 bits;
- memoria de datos RAM de 64 X 8 bits;
- capacidad E/S de 27 líneas;
- contador/temporizador de 8 bits.

Un tiempo de ciclo de 2,5 μs y un repertorio de 90 instrucciones con uno o dos ciclos de tiempo cada una, hace que el 8048, de pastilla única, tenga las mismas características que la mayoría de los microcomputadores con tecnología NMOS de varias pastillas, actualmente disponibles en el mercado, lo que hace que el 8048 sea un verdadero microordenador de "bajo costo". La necesidad de una única fuente de alimentación de 5 V para todos los componentes de la familia asegura que el "bajo costo" también sea aplicable a la fuente de alimentación del sistema.

Sin embargo, incluso con componentes de bajo costo, un proyecto puede resultar arriesgado debido a los elevados costos de desarrollo y rediseño resultantes de un diseño de producción inflexible. Se ha resuelto este problema mediante la creación de dos versiones del microordenador

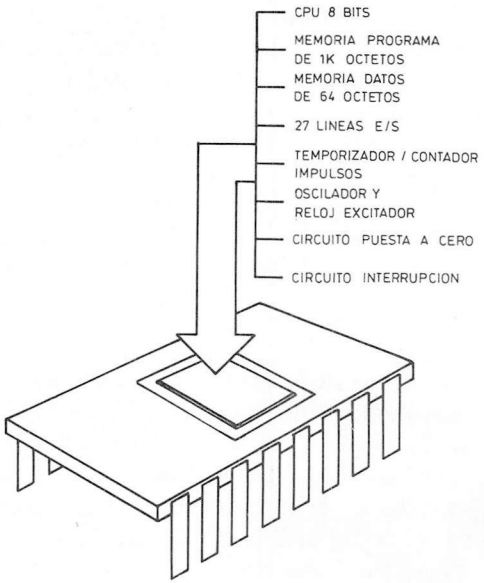


Fig. 7-73.–Características fundamentales del 8048.

8048, compatible patilla a patilla: el 8048 con memoria de programa ROM programable por máscara para aplicaciones de bajo costo, y el 8748 con memoria de programa EPROM, programable y borrable por el usuario para el desarrollo de prototipos. El 8748 es esencialmente un microordenador en una sola pastilla para montajes iniciales que pueden ser modificados una y otra vez durante su desarrollo y pre-producción, siendo después fácilmente reemplazable por el 8048 con ROM de bajo costo para producciones de gran volumen. El 8748 permite una fácil transición desde un desarrollo de prototipo a la producción del mismo, a la vez que permite un apoyo temporal a una producción mientras se fabrican las nuevas memorias ROM.

Para permitir que esta familia pueda resolver un amplio campo de problemas al diseñar un circuito y poder facilitar una futura ampliación, todas las funciones del 8048 pueden ser ampliadas externamente usando cualquiera de los periféricos y memorias disponibles. Es factible una eficaz ampliación de la capacidad de E/S a un bajo costo mediante el circuito 8243, el cual permite 16 líneas de E/S en un encapsulado de 24 pastillas. Para sistemas con gran necesidad de líneas E/S, se pueden utilizar varios circuitos 8243.

Para aplicaciones en teclados, visualizadores, líneas de transmisión en serie, etc., se pueden utilizar los periféricos de las familias 8080 y 2650. La memoria de datos y de programa se puede ampliar usando memorias normalizadas o las memorias 8355 y 8155, que incluyen también líneas de E/S programables y funciones temporizadoras.

El 8035 es un 8048 sin memoria de programa interna, que permite al usuario satisfacer sus necesidades de memoria de programa utilizando una gran variedad de memorias externas. El 8035 permite al usuario seleccionar su sistema con el mínimo costo y sin que importen las necesidades de memoria de programa.

Dentro de la familia MCS-48, existe una versión simplificada y por lo tanto más "económica", si cabe, del 8048, denominado 8021. Este microordenador contiene, en un encapsulado de sólo 28 patillas, la misma capacidad de memoria interna que el 8048, a la vez que utiliza su mismo repertorio de instrucciones. La capacidad de E/S queda reducida a 21 líneas, con posibilidad de fácil ampliación externa con circuitos TTL de bajo costo. Asimismo, contiene un circuito que detecta el "cruce por cero". Completan la familia MCS-48, cuatro microordenadores de reciente creación: el 8049 que es igual al 8048 pero con doble capacidad de memoria interna tanto RAM como ROM, el 8039 igual al 8035 pero con doble capacidad de la RAM interna, el 8022 y 8041 de características especiales.

Características principales

- Fuente de alimentación única de 5 V.
- Encapsulado DIP de 40 patillas.
- Compatibles patilla a patilla tanto la versión ROM como la EPROM.
- Ciclo de 2,5 μs.
- En todas las instrucciones un tiempo de ejecución de 1 o 2 ciclos como máximo.
- Dos bancos de registros de trabajo.
- Frecuencia de reloj controlada por cristal, por inductancia o bien generada exteriormente.
- Posibilidad de avance de programa "paso a paso".
- Ocho niveles de subrutinas.

La familia MCS-48

8048 Microordenador con ROM interna.
8748 Microordenador con EPROM interna.
8035 Microordenador sin memoria interna.
8021 Microordenador de bajo costo.
8243 Ampliador de E/S.
8355 Memoria ROM y ampliador de E/S.
8755 Memoria EPROM y ampliador de E/S.
8155 Memoria RAM y ampliador de E/S.
8049 Microordenador con 2 K de ROM interna.
8039 Microordenador con RAM interna de 128 palabras de 8 bits.
8022 Microordenador de gran integración.
8041 Microordenador para control de periféricos.

ARQUITECTURA Y CARACTERISTICAS FUNCIONALES

A continuación se describe cada uno de los bloques que componen el 8048 y algunas de las funciones que se pueden desarrollar (figura 7-74).

Sección aritmética

El bloque aritmético del procesador contiene todas las funciones básicas de tratamiento de datos y consta de las siguientes partes:

- Unidad aritmética y lógica, ALU (Arithmetic Logic Unit).
- Acumulador.
- Indicador de acarreo.
- Decodificador de instrucciones.

El acumulador es un registro a través del cual pasan todos los datos a tratar, procedentes de los registros de E/S, memoria y la ALU (operandos y resultados).

En la ALU se realizan todas las operaciones entre datos, tanto aritméticas (suma) como lógicas (AND, OR, NOR).

Memoria de Programa

La memoria de programa interna consta de 1024 palabras de 8 bits cada una, las cuales son direccionadas mediante el contador de programa.

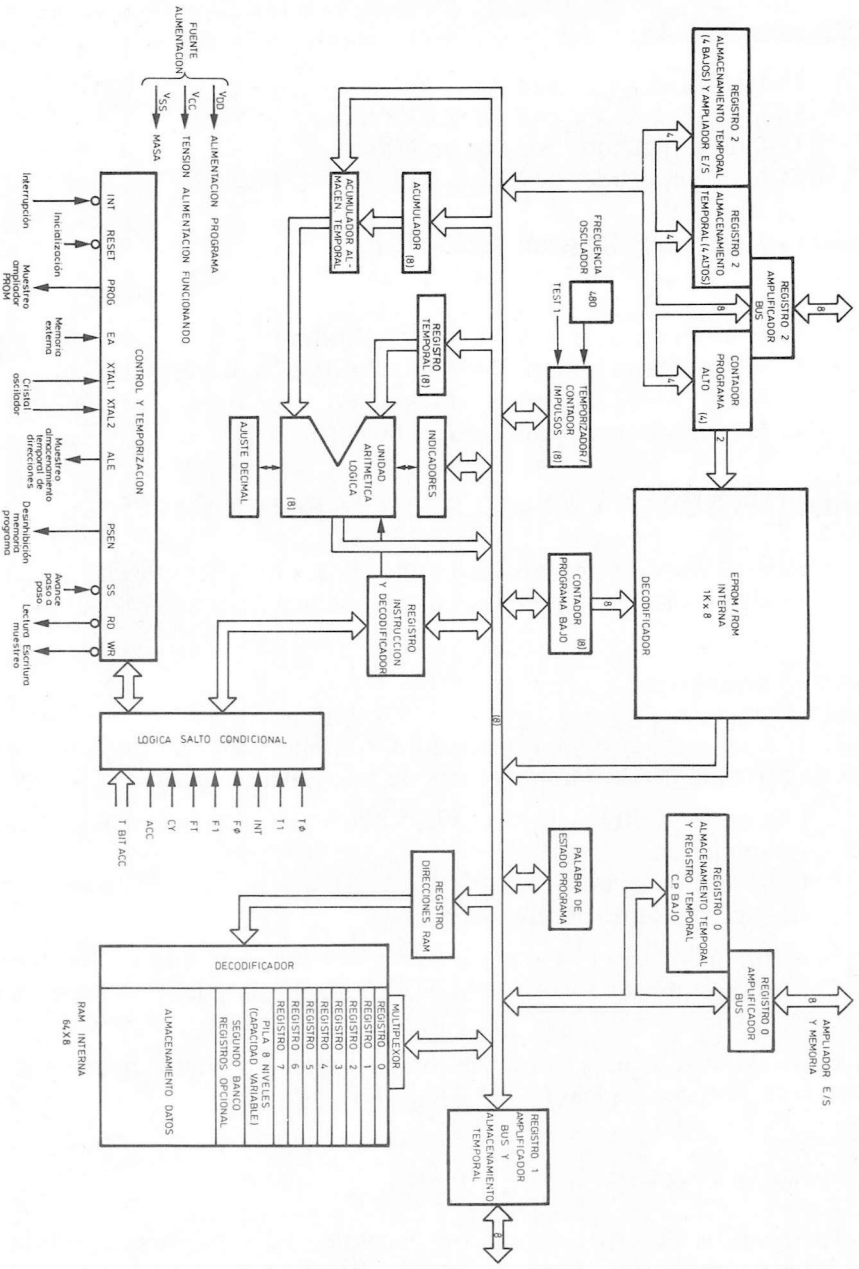


Fig. 7-74.–Diagrama por bloques del 8048.

En el 8748 esta memoria es una EPROM programable y borrable por el usuario y en el 8048 la memoria es una ROM programada mediante máscara en la fábrica. El 8035 no tiene memoria de programa interna y se utiliza con memorias externas. El código de programa es completamente intercambiable entre las tres versiones (fig. 7-75).

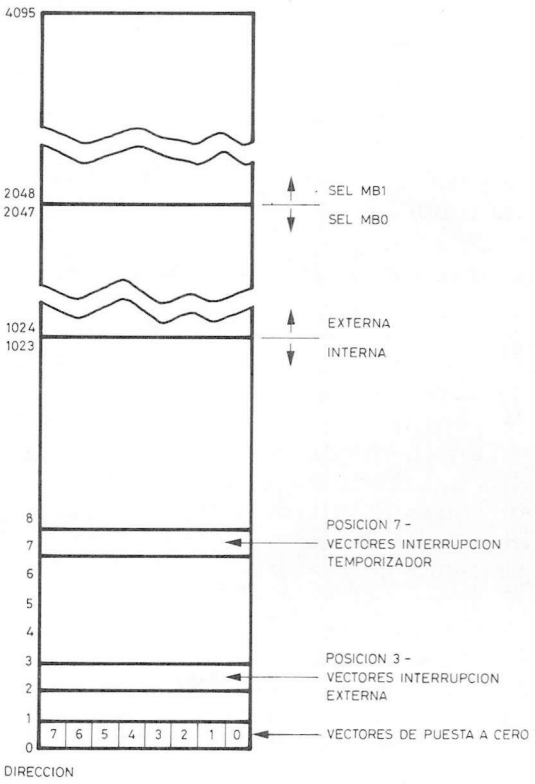


Fig. 7-75.—Mapa de la memoria de programa.

Hay tres posiciones en la memoria de programa de especial importancia:

Posición 0

Activa la puesta a cero del procesador, la primera instrucción se toma a partir de la posición 0.

Posición 3

Activa la entrada de interrupción del procesador (si el bit de interrupción está desinhibido), genera un salto a una posición de memoria.

Posición 7

Una interrupción provocada por el rebasamiento de la capacidad del temporizador/contador (si está desinhibido), provoca un salto a subrutina.

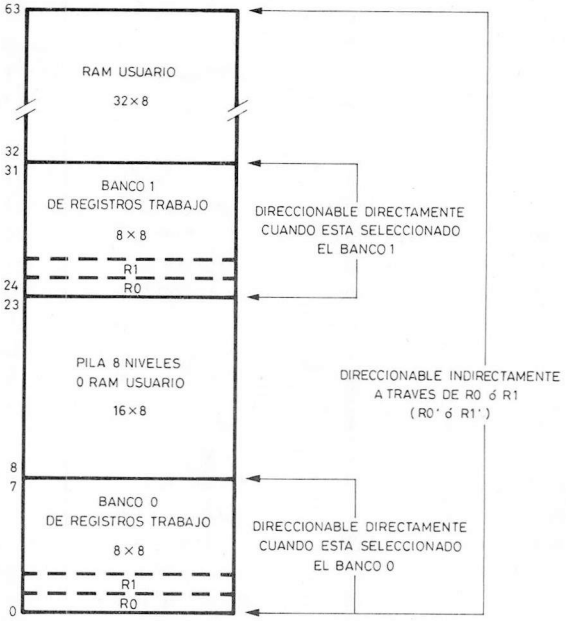
Es decir, la primera instrucción a ejecutar después de la inicialización es almacenada en la posición 0, la primera palabra de una subrutina de interrupción externa es almacenada en la posición 3 y la primera palabra de una subrutina de interrupción del temporizador/contador es almacenada en la posición 7. La memoria de programa se puede utilizar para almacenar constantes así como instrucciones de programa.

Memoria de datos

La memoria de datos interna está organizada en 64 palabras de 8 bits cada una. Cada una de las 64 posiciones de la memoria se direcciona indirectamente a través de dos registros punteros de la RAM que se encuentran en las direcciones 0 y 1 del banco de registros. Las primeras ocho posiciones de memoria (0-7) se designan como registros de trabajo, siendo direccionadas directamente mediante varias instrucciones. Ya que estos registros son fácilmente direccionables, se utilizan normalmente para almacenar resultados intermedios a los que se necesita acceder frecuentemente. La instrucción DJNZ permite efectuar lazos de retardo muy sencillos, mediante el decremento y posterior examen del contenido de un registro en una sola instrucción (fig. 7-76).

Mediante la ejecución de la instrucción de conmutación del banco de registro (SEL RB), las posiciones 24-31 de la RAM son designadas como registros de trabajo en lugar de las posiciones 0-7 que serán en este caso directamente direccionables como posiciones de memoria. Este segundo banco de registros de trabajo se puede utilizar como ampliación del primer banco o reservado para su uso durante una subrutina de interrupción, permitiendo reservar el contenido de los registros del banco 0, utilizados en el mismo programa, mediante la conmutación de banco. En caso de no ser utilizado este segundo banco, las posiciones 24-31 pueden ser direccionadas como aplicación general de memoria RAM.

Fig. 7-76.–Mapa de la memoria de datos.



Ya que los dos registros punteros de la RAM, R_0 y R_1, son una parte del sistema de registro de trabajo, la conmutación de banco crea dos nuevos registros punteros (R_0' y R_1'), los cuales se pueden utilizar junto con R_0 y R_1 para facilitar el acceso al mismo tiempo a cuatro zonas de trabajo separadas dentro de la RAM. Las posiciones de la RAM (8-23) tienen también una doble función ya que contienen la pila del contador de programa como se explicará más adelante. Estas posiciones se direccionan por la pila del puntero durante llamadas a subrutina, así como por los registros punteros de la RAM, R_0 y R_1. Si el nivel de subrutinas es inferior a 8, no es necesaria toda la pila de registros y puede ser utilizada como posiciones de memoria RAM de aplicación general. Cualquier nivel de subrutina no utilizado permite al usuario disponer de dos posiciones adicionales de memoria RAM.

Entrada/Salida

El 8048 tiene 27 líneas que se pueden utilizar como entrada o salida de datos. Estas líneas están agrupadas en 3 registros de E/S de 8 líneas

cada uno, teniendo funciones de entrada, salida o registro bidireccional, y 3 entradas de examen que pueden alterar secuencias de programa cuando son examinadas por instrucciones de salto condicional.

Registros de E/S 1 y 2

Los registros de E/S 1 y 2 constan de 8 bits cada uno y tienen ambos idénticas características. Los datos grabados en estos registros de E/S se almacenan estáticamente y permanecen invariables mientras no se graben otros datos. Actuando como registros de entrada, los datos no se almacenan y deben permanecer presentes mientras se efectúa la lectura de los mismos mediante una instrucción de entrada de datos. Las entradas son totalmente compatibles con TTL y las salidas son capaces de excitar una carga TTL.

Las líneas de los registros de E/S 1 y 2 se llaman "quasibidireccionales" debido a que el circuito de salida tiene una estructura especial que permite que cada línea pueda servir como entrada, salida y salida con almacenamiento temporal estático de los datos. La figura 7-77 muestra el esquema del circuito. Cada línea está conectada a + 5 V a través de un dispositivo resistivo de impedancia relativamente elevada (~ 50 kΩ). Esta conexión da la suficiente corriente como para excitar, en estado alto, un dispositivo TTL, a la vez que se puede llevar a estado bajo por una puerta TTL; esto permite que el mismo terminal se pueda utilizar como entrada y a la vez como salida. Para proporcionar tiempos de conmutación rápidos en las transiciones de "0" a "1", se conmuta rápidamente (~ 500 ns) un dispositivo de impedancia relativamente baja (~ 5 kΩ) cuando se graba un "1" en la línea. Cuando se graba un "0" en la línea, un dispositivo de baja impedancia (~ 300 Ω) proporciona una corriente con capacidad para excitar dispositivos TTL. Siempre que se quiera utilizar una línea como entrada se ha de grabar un "1" previamente en la misma. La puesta a cero inicializa todas las líneas a un estado de alta impedancia en estado "1". Esta estructura permite que un mismo terminal pueda actuar como entrada o salida y asimismo que pueda haber líneas de entrada y líneas de salida al mismo tiempo en un mismo registro de E/S.

El registro de E/S "quasibidireccional" en combinación con las instrucciones lógicas ANL y ORL proporciona un medio eficaz de manejo de líneas individuales en un procesador de 8 bits.

Barra de datos o bus de datos

La barra de datos es también un registro de E/S de 8 bits bidireccional, con entradas y salidas de muestreo (*strobe*) comunes. Si la característica

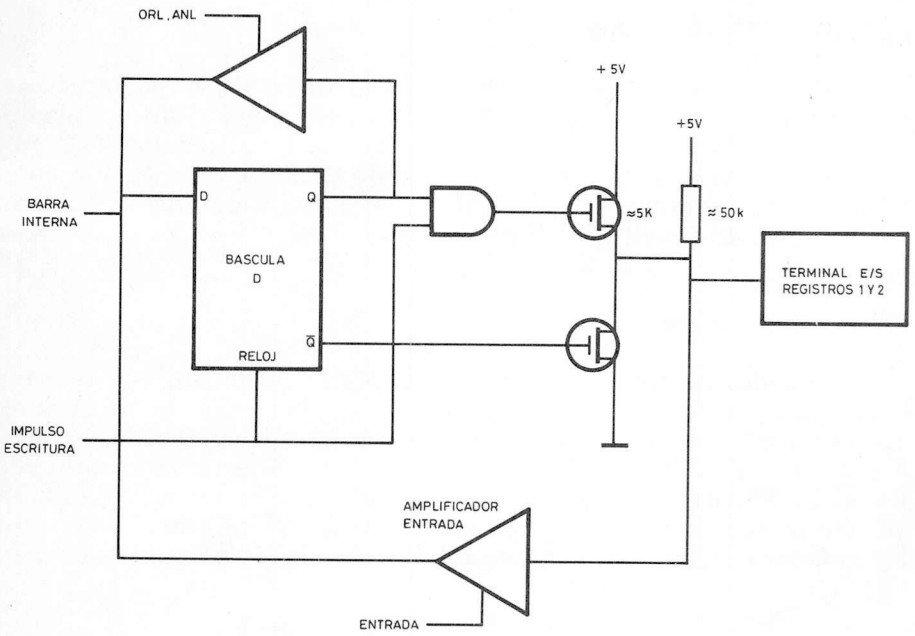


Fig. 7-77.–Estructura de registro quasibidereccional.

bidireccionalidad no es necesaria, la barra de datos puede comportarse como un registro de E/S con salidas con almacenamiento temporal estático o entradas sin almacenamiento. Las líneas de entrada y las de salida no pueden mezclarse, es decir, las 8 líneas del registro de E/S tienen que ser todas de salida o bien de entrada.

Como registro de E/S que es, los datos se escriben y se almacenan mediante la instrucción OUTL y se introducen mediante la instrucción INS. Las instrucciones INS y OUTL generan impulsos en las correspondientes líneas de muestreo de salida: RD (lectura) o WR (escritura); no obstante, en el modo de funcionamiento de registro de E/S estático generalmente no se utilizan. Como registro de E/S bidireccional, la instrucción MOVX se emplea para leer el registro de E/S. Una escritura en el registro de E/S genera un impulso en el terminal de salida WR que hace válido el dato de salida a partir del flanco inicial de WR. Una lectura en el registro de E/S genera un impulso en el terminal de salida RD, haciendo válido el dato de entrada en el momento en que se produce el primer flanco en RD. Cuando no se escribe o se lee, la barra de datos está en estado de alta impedancia.

Entradas INT y de prueba

Existen tres terminales que sirven como entradas y a la vez pueden ser examinadas mediante instrucciones de salto condicional. Estos terminales son T_0, T_1 e INT. Mediante estos terminales se consiguen datos de programa sin necesidad de cargar el contenido de un registro de E/S en el acumulador. Asimismo pueden realizar otras funciones que se explican en el aparato de descripción de patillas.

Contador de programa y pila

El contador de programa es un contador independiente mientras que la pila del contador de programa se implementa por medio de pares de registros en la memoria de datos. Unicamente se utilizan 10 bits del contador de programa para direccionar las 1024 palabras de la memoria de programa interna, mientras que los dos bits más significativos se utilizan cuando se debe trabajar con memoria de programa externa. El contador de programa se pone a cero cuando se activa la línea de puesta a cero (RESET).

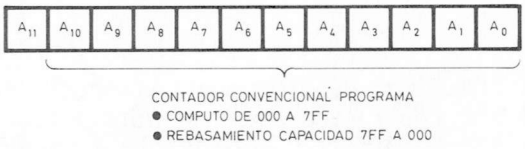


Figura 7-78.–Contador de programa.

Una interrupción o salto a subrutina hace que el contenido del contador de programa sea almacenado en uno de los 8 pares de registros de la pila de contador de programa. El par viene determinado por un puntero de pila de 3 bits que forma parte de la palabra de estado, PSW (Program Status Word). Las posiciones 8 a 23 de la memoria de datos RAM son utilizables como registros de pila y para almacenar el contador de programa y 4 bits de la PSW. Cuando el puntero de la pila está en la posición 000, direcciona las posiciones 8 y 9 de la RAM. El primer salto a subrutina o interrupción produce una transferencia del contenido del contador de programa hacia las posiciones 8 y 9 de la RAM. El puntero de pila se incrementa en una unidad y queda apuntado a las posiciones 10 y 11 de la RAM antes de que se produzca otra llamada a subrutina (CALL). Se pueden ejecutar unas subrutinas dentro de otras hasta un tope máximo de 8 veces sin que se rebase la capacidad de la pila.

El final de una subrutina, señalado por las instrucciones de retorno (RET o RETR), produce un decremento en el puntero de pila y transfiere el contenido del par de registros al contador de programa.

Palabra de Estado de Programa (PSW)

Existen una serie de biestables organizados en una palabra de 8 bits, los cuales pueden cargarse desde o al acumulador y que se encargan de informar del estado de ciertos parámetros de un programa. Esta palabra de 8 bits es lo que constituye la llamada palabra de estado de programa, PSW (Program Status Word). Estos biestables pueden ser leídos o grabados muy fácilmente. La figura 7-79 muestra la información disponible en la palabra.

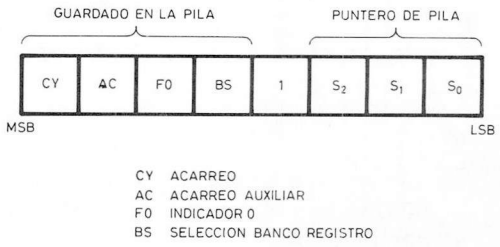


Fig 7-79.—Palabra de estado de programa PSW.

Los cuatro bits más significativos de la PSW son almacenados en la pila del contador de programa al producirse un salto a subrutina o vector de interrupción y restablecidos a su estado inicial automáticamente mediante la instrucción RETR. La instrucción de retorno RET no establece la PSW.

La función de cada uno de los bits de la PSW es la siguiente:

Bits 0-2: bits del puntero de pila (S_0, S_1, S_2).

Bit 3: no se usa (se encuentra a nivel "1" para casos de lectura de la PSW).

Bit 4: conmutador de banco de registros de trabajo.

Bit 5: bit del indicador 0 (F0), controlable por el usuario que puede ser borrado, complementado, o examinado mediante la instrucción de salto condicional JFO.

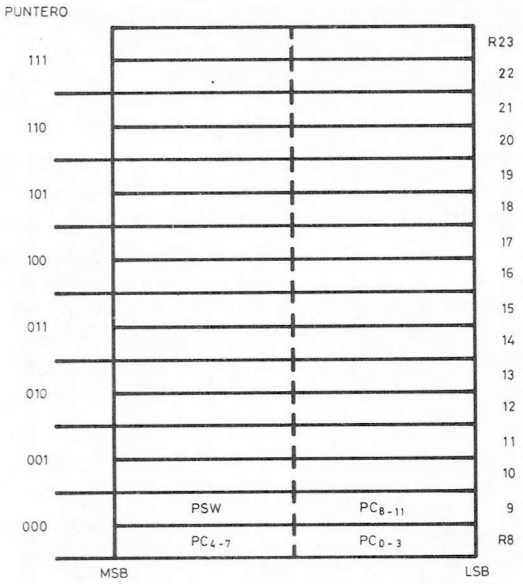


Fig 7-80.–Pila del contador de programa.

Bit 6: acarreo auxiliar, AC (Auxiliary Carry), generado mediante una instrucción de suma ADD y utilizado por la instrucción de ajuste decimal DA A.

Bit 7: acarreo, CY (Carry) que indica que una operación previa ha tenido un resultado con rebasamiento de capacidad en el acumulador.

Interrupción (fig. 7-81)

Al aplicar un nivel "0" en el terminal de entrada INT, se inicia una secuencia de interrupción. Se genera una interrupción mediante niveles 0 para permitir la conexión de diversas fuentes de interrupción en el terminal de entrada mediante el sistema de "puerta OR cableada". La línea de interrupción se muestrea en cada ciclo de máquina en el tiempo que dura la señal ALE y cuando se detecta una interrupción se produce un "salto a subrutina" a la posición 3 de la memoria de programa, en cuanto hayan sido completados todos los ciclos de la instrucción que se lleve a cabo en ese momento. Al igual que en cualquier llamada a subrutina, el contenido del contador de programa y la palabra de estado son guardados en la pila. Esta operación se describió en el apartado que trata del contador de pro-

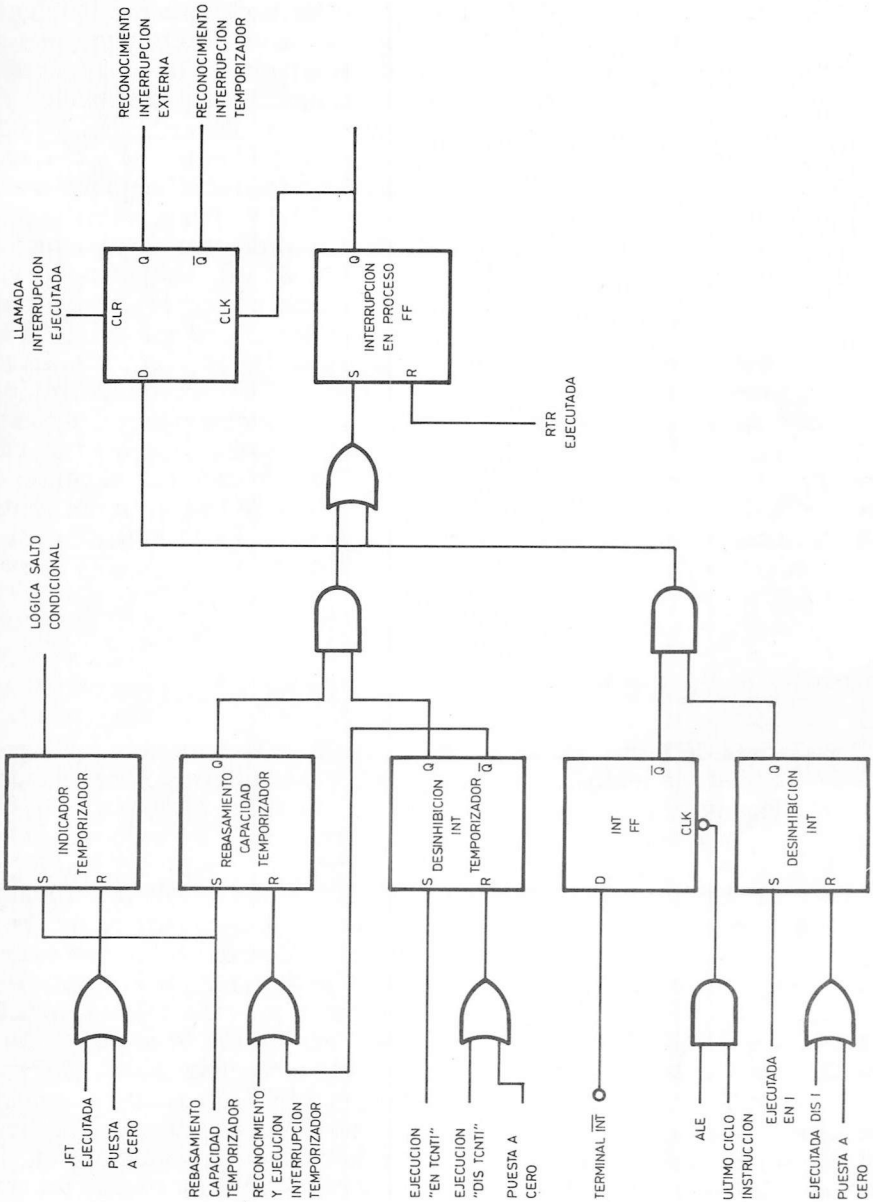


Fig. 7-81.–Lógica de interrupción.

grama y pila. La posición 3 de la memoria de programa normalmente contiene un salto incondicional a una subrutina de interrupción que se encuentra en otra parte de la memoria de programa. El final de una subrutina de interrupción se produce cuando es ejecutada la instrucción de retorno y restablecimiento del estado de programa RETR. El sistema de interrupciones consta de un único nivel en el cual, mientras se está ejecutando alguna interrupción, una nueva petición exterior es ignorada mientras no se ejecute la instrucción de retorno RETR que se encargará de desinhibir el circuito lógico de entrada de interrupciones. Esto ocurre al comienzo del segundo ciclo de la interrupción RETR. Esta secuencia es la misma para interrupciones internas producidas por el rebasamiento de la capacidad del temporizador. En caso de detectar una interrupción interna y otra externa al mismo tiempo, tendrá prioridad la externa. La interrupción del temporizador se describirá en el apartado temporizador/contador. Si se necesita una segunda fuente de interrupción externa se puede crear mediante la desinhibición del temporizador/contador, cargando el contador con la información EFE (uno menos que el cómputo máximo del contador), y desinhibiendo el modo de cómputo del contador de sucesos. Un flanco de bajada en la entrada T1 producirá un vector de interrupción hacia la posición 7 de la memoria.

Secuencia de interrupción

La entrada de interrupción puede ser inhibida o desinhibida por programa mediante la instrucción EN I y DIS I. Las interrupciones pueden ser inhibidas mediante la puesta a cero y mantenidas así hasta ser desinhibidas por programa. Una petición de interrupción debe ser retirada antes de que se ejecute la instrucción de retorno de subrutina RETR, de otra forma el procesador volvería a reiniciar la subrutina de interrupción inmediatamente. Muchos periféricos evitan esta situación mediante el borrado de la línea de petición de interrupción siempre que el procesador acceda (lectura o escritura) a los registros periféricos. Si el dispositivo que genera la interrupción no es accedido por el procesador, una línea de salida del 8048 puede ser designada como "aceptación de interrupción", la cual es activada por la propia subrutina de interrupción con el fin de borrar la petición de interrupción. El terminal INT puede ser examinado mediante la instrucción de salto condicional JNI. Esta instrucción puede ser usada para detectar la presencia de interrupciones pendientes de ejecución antes de que las mismas sean desinhibidas. En caso de que las interrupciones estén inhibidas, el terminal INT puede ser usado como entrada de muestreo al igual que T0 y T1.

Temporizador-Contador (fig. 7-82)

El 8048 contiene un contador que ayuda al usuario en el cómputo de sucesos externos y la generación de tiempos de retardo exactos sin recargar al procesador con la realización de estas funciones. En ambos modos de funcionamiento el contador se comporta de igual forma, la única diferencia consiste en la fuente de entrada al contador.

Contador

El contador binario de 8 bits puede ser cargado y leído mediante dos instrucciones MOV que transfieren el contenido del acumulador al contador y viceversa. El contenido del contador no queda afectado por la puesta a cero general, siendo inicializado únicamente mediante la instrucción MOV T, A. El contador puede pararse mediante la puesta a cero general o la instrucción STOP TCNT, manteniéndose parado mientras no es puesto en marcha como temporizador mediante la instrucción START T, o como contador de sucesos mediante la instrucción START CNT. Una vez puesto en marcha, el contador irá incrementando su valor hasta el máximo cómputo (FF), dando señal de rebase de capacidad y continuando el cómputo a partir de cero hasta que es parado mediante la instrucción STOP TCNT o la puesta a cero general.

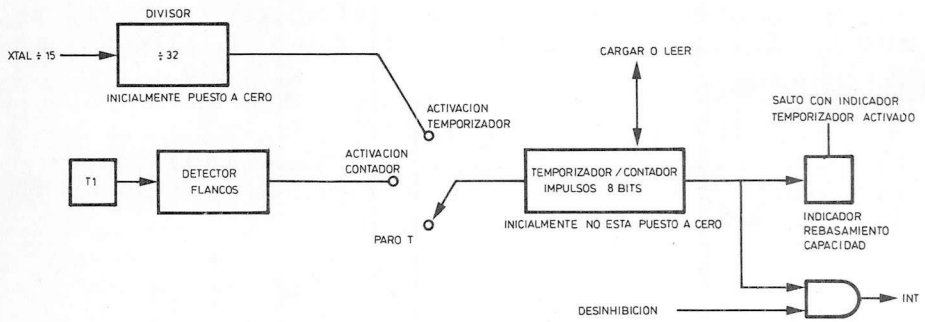


Fig. 7-82.–Temporizador/contador de sucesos.

El incremento desde el máximo valor de cómputo a cero (rebase de capacidad) activa un biestable indicador de rebase de capacidad y genera una petición de interrupción. El estado del indicador de rebase de capacidad puede examinarse mediante la instrucción de salto condicional JTF.

El indicador es borrado mediante la puesta a cero general o al ejecución de la instrucción JTF. La petición de interrupción es almacenada en un biestable y seguidamente combinada con la señal de petición de interrupción externa INT, a través de una puerta OR. La interrupción del temporizador puede ser desinhibida o inhibida independientemente de la señal de interrupción externa, mediante las instrucciones EN TCNTI y DIS TCNTI. Si está desinhibida, un rebasamiento de la capacidad en el contador causará un salto a la posición 7 de la memoria donde se encuentra almacenada la rutina de servicio del temporizador o del contador de sucesos. En caso de presentarse simultáneamente dos peticiones de interrupción, una externa y otra del temporizador, tendrá preferencia la externa y la llamada a subrutina de interrupción se hará en la posición 3 de la memoria. No obstante, la interrupción del temporizador es almacenada temporalmente, permaneciendo a la espera de que haya sido servido el dispositivo externo que pidió interrupción; en el momento que se produzca el retorno de la subrutina de interrupción externa, entrará inmediatamente en funcionamiento la subrutina de interrupción del temporizador. La petición de interrupción pendiente se borra en el momento que se salta a la posición 7 de la memoria, o puede ser suprimida mediante la instrucción DIS TCNTI.

Contador utilizado como contador de sucesos.

La ejecución de la instrucción START CNT conecta al terminal de entrada T1 a la entrada del contador y la desinhibe. Cada flanco de bajada en T1 originará un incremento de valor en el contador. La máxima velocidad a la cual el contador puede ser incrementado es una vez por cada tres ciclos (cada 7,5 μs si se usa un cristal de 6 MHz), no obstante ésta no es la mínima frecuencia a la que puede funcionar. La entrada T1 necesita estar un mínimo de 500 μs en estado alto después de cada transición.

Contador utilizado como temporizador

La ejecución de la instrucción START T conecta un reloj interno a la entrada del contador y desinhibe el contador. El reloj interno es el resultado de pasar los 400 kHz de ciclo de reloj básico (señal ALE) a través de un divisor por 32. El divisor se pone a cero durante la ejecución de la instrucción START T. El reloj resultante de 12,5 kHz incrementa el contador cada 80 μs (siempre que se encuentre conectado un cristal de 6 MHz). Se obtienen diferentes retardos, desde 80 μs hasta 20 ms (255 cómputos), cargando el contador con un valor (inicialización del contador) y la

detección de la señal de rebase de capacidad. Se pueden obtener tiempos de retardo mayores de 20 ms mediante la acumulación de múltiples impulsos de rebase de capacidad en un registro, controlando el proceso por programa. Para tiempos menores de 80 μs, puede aplicarse un reloj externo a la entrada T1 haciendo operar el contador en el modo "contador de sucesos". La señal ALE dividida por 3 o más puede hacer las funciones de reloj externo. Pueden obtenerse fácilmente tiempos de retardo muy pequeños o muy grandes mediante lazos de retardo realizados por programa.

Circuitos de reloj y de temporización (fig. 7-83)

La generación de secuencias en el 8048 está totalmente resuelta dentro del mismo, a excepción de la frecuencia de referencia que puede ser generada por cristal, por inductancias o por una fuente de reloj externa. Los circuitos de reloj y de temporización pueden dividirse en los siguientes bloques funcionales:

Oscilador

El oscilador consta de una serie de circuitos resonantes de alta ganancia con una escala de frecuencias que va de 1 a 6 MHz. El terminal exterior X1 es la entrada a la etapa amplificadora, siendo X2 la salida de la misma. Un cristal o inductancia conectada entre los terminales X1 y X2 proporciona la realimentación y desplazamiento de fase necesarios para que el circuito se ponga a oscilar. Un cristal de 5.9904 MHz permite obtener fácilmente todas las frecuencias normalizadas en comunicaciones. Si no es necesaria una gran precisión en la frecuencia de referencia, ni una alta velocidad en el procesador, puede utilizarse una inductancia en lugar de un cristal. Con una inductancia, la frecuencia del oscilador puede ser aproximadamente de 3 a 5 MHz. Para mayores velocidades de operación se utilizará un cristal. Asimismo, se puede aplicar un reloj externo a los terminales X1 y X2 como fuente de frecuencia.

Contador de estados

La salida del oscilador se divide por tres en el contador de estados para crear un reloj que define el tiempo de los estados de la máquina (CLK). La señal CLK puede estar disponible en el terminal externo T0 mediante la ejecución de la instrucción ENTO CLK. La salida CLK por T0 puede ser inhibida mediante la puesta a cero general del procesador.

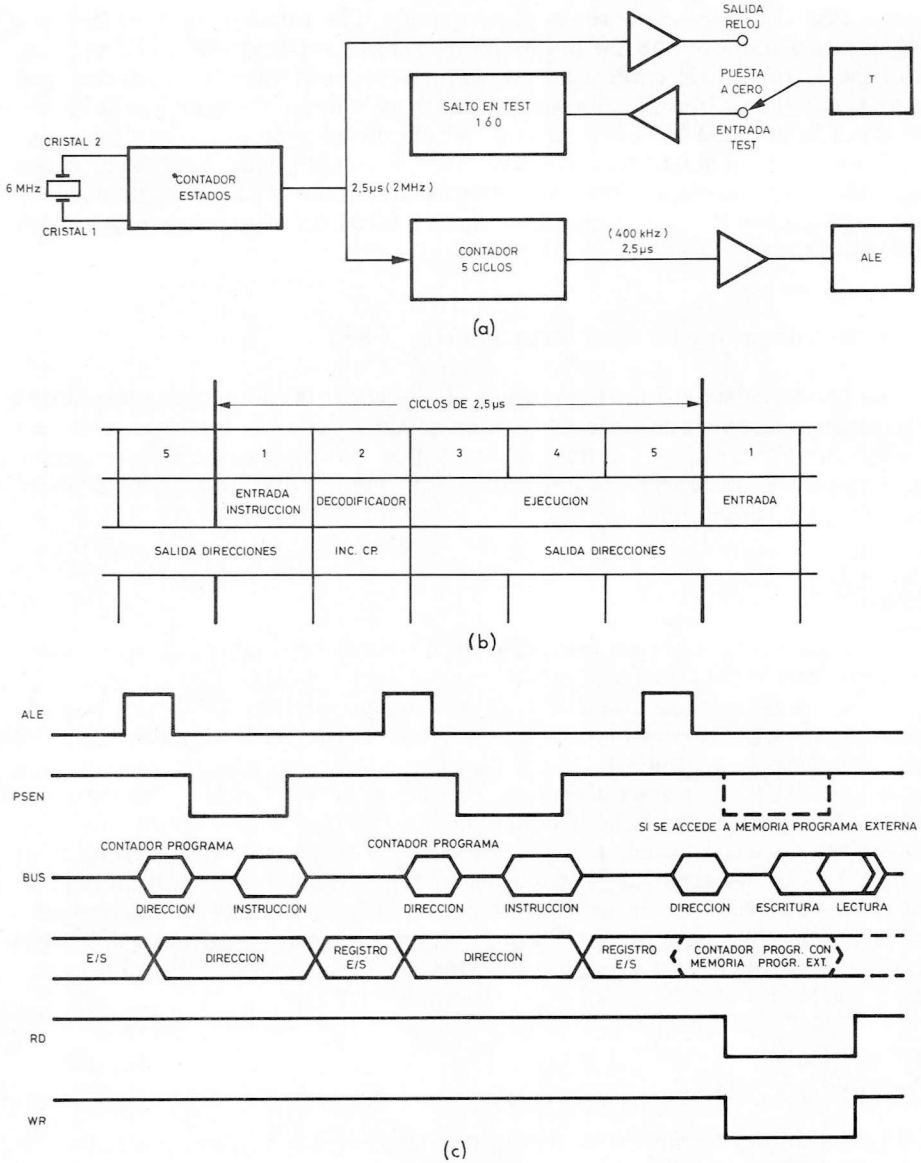


Fig. 7-83.–(*a*) Diagrama de reloj del 8048. (*b*) Ciclo de instrucción. (*c*) Temporización del ciclo para memoria externa.

Contador de ciclos

La señal CLK es dividida posteriormente por cinco en el contador de ciclos, proporcionando una señal de reloj que define el ciclo de máquina compuesto por cinco estados de máquina. Este reloj se demonina ALE (Address Latch Enable) y, entre otras funciones, se utiliza para desinhibir los registros de dirección de las memorias externas. Esta señal se encuentra disponible permanentemente en el terminal de salida ALE.

Puesta a cero (figuras 7-84 y 7-85)

La entrada de puesta a cero proporciona un medio para poder inicializar el procesador. Consta de un disparador de Schmitt y una resistencia conectada a positivo que, en combinación con un condensador exterior de 1 μF, proporciona un impulso de puesta a cero interno de suficiente duración como para garantizar que todo el circuito ha sido puesto a cero. En caso de que el impulso de puesta a cero sea generado exteriormente, el terminal de puesta a cero tiene que mantenerse a nivel de masa (0,5 V máximo) por un período de tiempo mínimo de 50 ms inmediatamente después de que la tensión de alimentación ha llegado a su valor nominal.

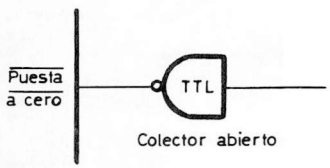


Fig. 7-84.– Puesta a cero exterior.

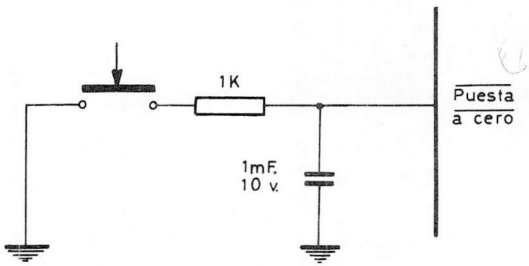


Fig. 7-85.–Restablecimiento de la alimentación.

La puesta a cero realiza las siguientes funciones:

- Pone a cero el contador de programa.
- Pone a cero el puntero de pila.
- Selecciona el banco 0 de registros.
- Selecciona el banco 0 de memoria.

- Pone la barra en el estado de alta impedancia (excepto cuando EA = = 5 V).
- Pone los registros de E/S 1 y 2 en el modo de entrada.
- Inhibe las interrupciones (la de temporizador y la externa).
- Para el temporizador.
- Borra el indicador de temporizador (FT).
- Borra F0 y F1.
- Inhibe la salida de reloj desde T0.

Avance paso a paso

Esta característica proporciona al usuario la posibilidad de hacer progresar el programa instrucción a instrucción. Mientras el programa se encuentra detenido, la dirección de la siguiente instrucción a ejecutar se encuentra disponible en la barra y la mitad del registro de E/S 2. El usuario puede, gracias a esto, seguir el programa a través de cualquier paso de instrucción. En la figura 7-86 puede verse un diagrama de temporización mostrando la relación entre la salida ALE y la entrada SS. El contenido del amplificador de la barra se pierde durante el avance de un paso de programa; no obstante, se puede añadir un registro para recuperar lo perdido en caso necesario.

Secuencia

El 8048, en avance paso a paso opera de la forma siguiente:

- El procesador recibe una petición de paro mediante la aplicación de un nivel bajo en SS.
- El procesador responde deteniéndose en el tiempo de búsqueda de la siguiente instrucción; Si se está ejecutando una instrucción de dos ciclos cuando se recibe la orden de avance paso a paso, ambos ciclos son completados antes de parar el programa.
- El procesador se da por enterado del estado de paro mediante la subida a uno de ALE. En este estado (que puede ser mantenido indefinidamente), la dirección de la siguiente instrucción a ejecutar se encuentra presente en la barra y en la mitad del registro de E/S 2.
- Posteriormente, la entrada SS pasa a nivel uno lógico, reanudando su funcionamiento el procesador para poder buscar la siguiente instrucción a ejecutar. La reanudación del funcionamiento es indicada por el procesador poniendo a nivel cero lógico la salida ALE.

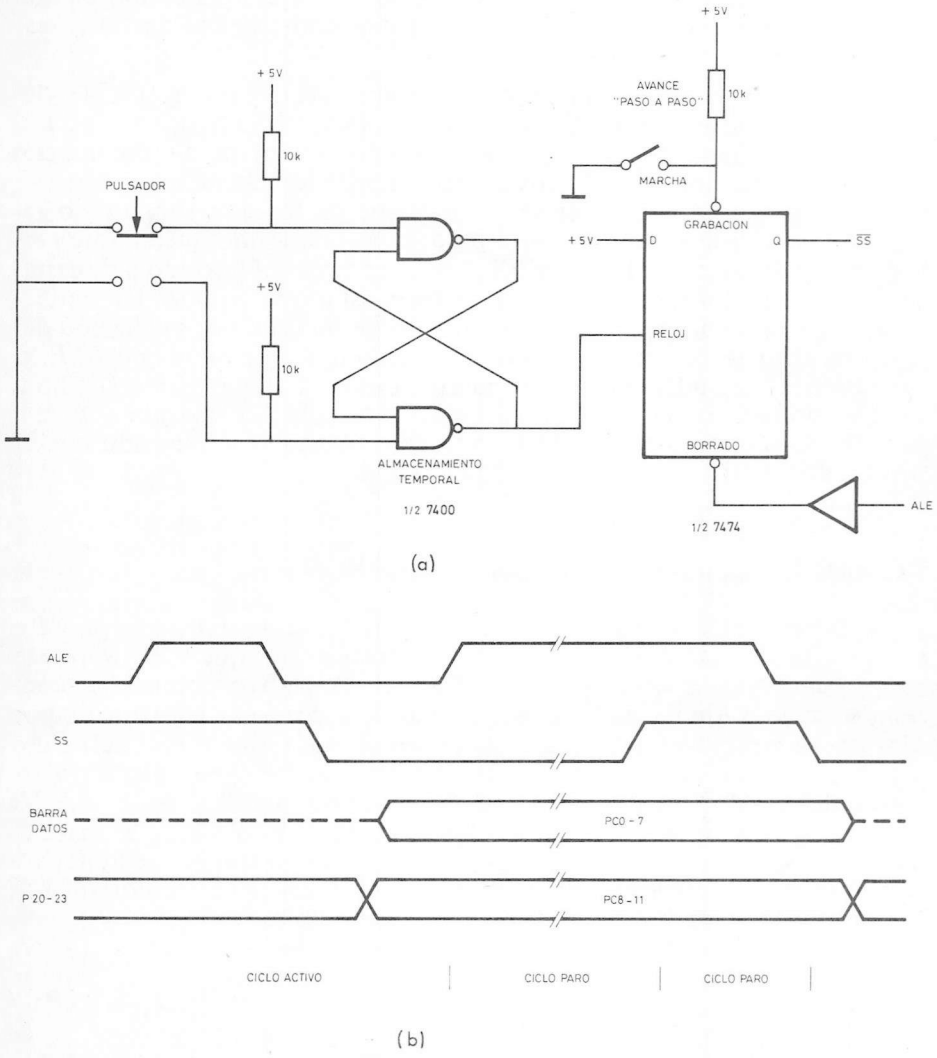


Fig. 7-86.– (*a*) Circuito de paso a paso. (*b*)Secuencia de paso a paso.

– Para parar el procesador en la siguiente instrucción, SS debe llevarse a cero otra vez tan pronto como ALE pase a cero. En caso de que SS permanezca en estado alto, el procesador seguirá en funcionamiento.

El diagrama de funcionamiento del avance paso a paso en el 8748 viene mostrado en la figura 7-86. Para generar la señal SS se utiliza un biestable tipo D con borrado y puesta a uno. En el modo de marcha (procesador funcionando) la entrada SS permanece a nivel uno ya que se mantiene la puesta a uno del biestable (tiene preferencia sobre el borrado). Para seguir avanzando paso a paso, la puesta a uno queda libre permitiendo a la señal ALE poner SS a cero a través del borrado del biestable. El procesador se encuentra ahora en el estado de parado. La siguiente instrucción es iniciada por la aplicación de un uno, por mediación del reloj en el biestable. Este uno no aparecerá en SS a menos que ALE lo sea también, dejando libre el borrado del biestable. En el cambio de SS a uno, el procesador comienza una instrucción a la vez que pone ALE a cero, borrando SS a través de la entrada de borrado y haciendo que el procesador vuelva a entrar en estado de parado.

Régimen de baja potencia (sólo para el 8048 ROM)

En la versión 8048 ROM se ha añadido un circuito extra que permite suprimir la alimentación de todo el circuito mientras que la RAM puede seguir funcionando a baja potencia. En régimen de baja potencia, el contenido de la RAM de datos puede mantenerse con sólo un 10 a 15 por 100 de las necesidades normales de potencia. La tensión V_{CC} actúa como alimentación de todo el circuito, mientras que V_{DD} alimenta únicamente la RAM. Operando normalmente, ambas patillas están a 5 V, mientras que en estado de reposo V_{CC} está a masa y sólo V_{DD} se mantiene a 5 V. Aplicando la puesta a cero al procesador, se inhibe cualquier acceso a la RAM y garantiza que la RAM no sea alterada involuntariamente por la supresión de V_{CC}.

Descripción de patillas (fig. 7-87)

El encapsulado del 8048 y 8748 es de 40 patillas dispuestas en doble línea (DIP). En la tabla 1 se da el resumen de las funciones de cada una de las patillas. En alguna de las patillas se explica la función concreta de cada una de ellas en un sistema ampliado mediante componentes externos. A no ser que se indique lo contrario, cada entrada es compatible con dispositivos TTL y cada salida puede excitar una carga TTL.

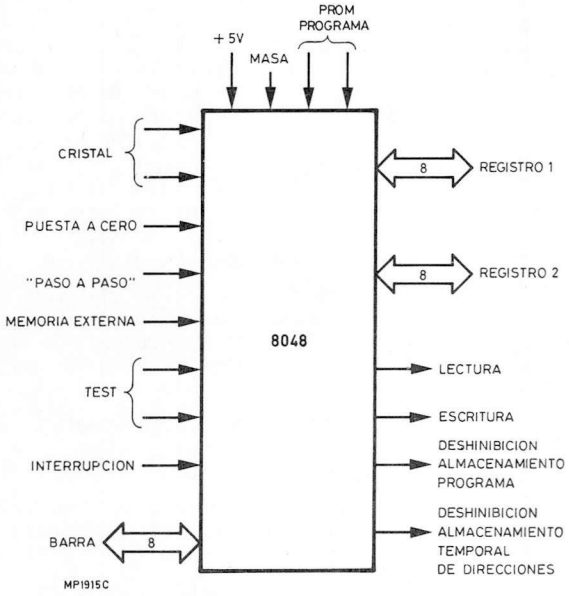


Fig. 7-87.–Símbolo lógico del 8048.

TABLA 1

Designación	Patilla	Función
V_{SS}	20	Masa.
V_{DD}	26	Tensión de alimentación para programación; +25 V programando, +5 V operando tanto en ROM como PROM. Patilla de régimen de baja potencia en la versión 8048 ROM.
V_{CC}	40	Tensión de alimentación; +5 V operando y programando cuando se trata del 8748.
PROG	25	Patilla de entrada de impulsos de programación (+25 V) durante la programación del 8748. E/S del 8243.
P10-P17 (Registro de E/S 1)	27-34	Registro de E/S quasibidireccional de 8 bits.
P20-P27 (Registro de E/S 2)	21-24	Registro de E/S quasibidireccional de 8 bits.
	35-38	P20-P23 contienen los cuatro bits altos del contador de programa durante un acceso a memoria externa y sirven como barra de datos de E/S para el 8243.

Designación	Patilla	Función
D0-D7 (BUS)	12-19	Registro de E/S bidireccional que puede ser leído y escrito en sincronismo mediante las salidas de muestreo RD y WR. El registro de E/S puede almacenar estáticamente una información. Contiene los 8 bits bajos del contador de programa durante un acceso a memoria externa y recibe la instrucción direccionada bajo el control de la señal PSEN. También contiene la dirección y el dato durante la ejecución de una instrucción de almacenamiento de información en RAM externa, bajo el control de las señales ALE, RD y WR.
T0	1	Terminal de entrada examinable mediante las instrucciones condicionales de transferencia JT0 y JNT0. T0 puede ser designado como salida de reloj usando la instrucción ENTO CLK. T0 se usa también durante la programación de la EPROM (8748).
T1	39	Terminal de entrada examinable mediante las instrucciones JT1 y JNT1. Asimismo, puede designarse como entrada del contador de sucesos mediante la instrucción STRT CNT.
INT	6	Entrada de interrupción. Inicia una interrupción siempre que éstas estén desinhibidas. Las interrupciones son inhibidas después de aplicar la puesta a cero. (Nivel activo el cero.)
RD	8	Salida de muestreo activada durante una lectura del BUS. Puede usarse para desinhibir la información de entrada al BUS, proveniente de un dispositivo exterior. (Nivel activo el cero.) Se utiliza como muestreo de lectura de memoria de datos externa.
RESET	4	Entrada de puesta a cero para inicializar el procesador. También se utiliza durante la programación y verificación de la EPROM del 8748. (Nivel activo el cero.)
WR	10	Salida de muestreo, activa durante una escritura del BUS. (Nivel activo el cero.) Se usa como muestreo de escritura a memoria de datos externa.
ALE	11	Desinhibición del registro de almacenamiento temporal de las direcciones. Esta señal se presenta durante cada ciclo y es utilizable como salida de reloj. El flanco de bajada de ALE introduce la dirección dentro de una memoria de datos o de programa externa.
PSEN	9	Desinhibición del almacenador del programa. Esta salida se presenta sólo durante un acceso a memoria de programa externa. (Nivel activo el cero.)
SS	5	Entrada de avance paso a paso que se usa, junto a la señal ALE, para el procesador avance "un solo paso" en cada instrucción. (Nivel activo el cero.)
EA	7	Entrada de acceso externo que permite buscar el programa en la memoria externa. Es importante en simuladores y programas correctores y esencial para examen y verificación de un programa. (Nivel activo el uno.)

Designación	Patilla	Función
XTAL 1	2	Entrada 1 del cristal para el oscilador interno o entrada para una fuente de frecuencia externa.
XTAL 2	3	Entrada 2 del cristal.

CONJUNTO DE INSTRUCCIONES

Las instrucciones tienen dos octetos como máximo y aproximadamente el 70 por 100 tienen un solo octeto de longitud. Además, todas las instrucciones tienen uno o dos ciclos de tiempo de ejecución (2,5 μs o 5 μs si usamos un cristal de 6 MHz), y aproximadamente el 50 por 100 tienen un solo ciclo. Las instrucciones de doble ciclo incluyen todas las instrucciones inmediatas y las de E/S.

El 8048 puede efectuar operaciones aritméticas, tanto en código binario como en BCD.

Transferencia de datos

Tal como se puede observar en la figura 7-88, el acumulador de 8 bits es el punto de paso obligado para todas las transferencias de datos en el 8048. Los datos pueden ser transferidos directamente entre los 8 registros de cada banco de registros de trabajo y el acumulador, el registro de destino es especificado por la propia instrucción. El conjunto de posiciones de memoria RAM interna está organizado como si se tratara de una memoria de datos, y es direccionada indirectamente a través del contenido de los registros R0 y R1 incluidos en los bancos de registro de trabajo. Estos registros también son usados para direccionar indirectamente una memoria de datos externa en caso de que exista. Las transferencias hacia y desde la RAM interna requieren un solo ciclo, mientras que con la RAM externa se necesitan dos. Las constantes almacenadas en la memoria de programa pueden ser cargadas directamente en el acumulador y en los 8 registros de trabajo. Los datos pueden ser transferidos directamente entre el acumulador y el temporizador/contador interno, o entre el acumulador y la palabra de estado (PSW). Modificando la PSW se puede alterar el estado de la máquina y restablecerla después de una interrupción o de una alteración del puntero de la pila, si se considera necesario.

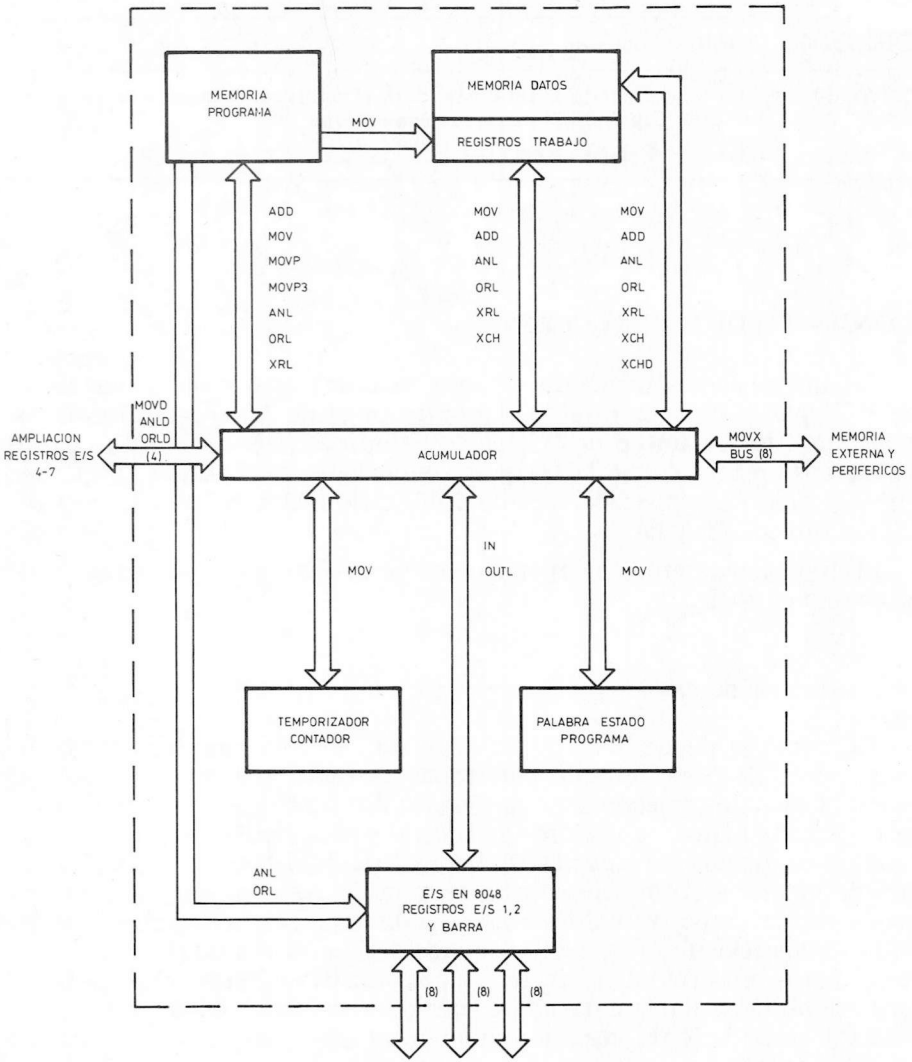


Fig. 7-88.–Instrucciones de transferencia de datos.

Operaciones con el acumulador

A través del acumulador pueden ser sumados, con o sin acarreo, datos inmediatos, de memoria, o de registros de trabajo. Asimismo, estos datos

pueden ser operados, mediante el acumulador, por funciones AND, OR y OR exclusivo. Los datos pueden ser transferidos hacia o desde el acumulador, a partir de registros de trabajo, o de la memoria de datos. Los dos contenidos pueden ser cambiados en una sencilla operación.

Además, los cuatro bits menos significativos del acumulador pueden ser intercambiados con los cuatro bits menos significativos de cualquier posición en la memoria RAM de datos interna. Esta instrucción, junto con una instrucción que se encarga de permutar los 4 bits menos significativos con los 4 bits más significativos del acumulador, permite una fácil manipulación de las palabras de 4 bits, incluyendo números en código BCD. Para facilitar operaciones aritméticas en código BCD existe la instrucción de ajuste decimal. Esta instrucción es utilizada para corregir el resultado de una operación de suma binaria entre dos números en código BCD. Ejecutando el ajuste decimal en el resultado binario de una operación se obtiene el resultado equivalente en código BCD.

Finalmente, el acumulador puede ser incrementado, decrementado, borrado, complementado, y desplazado 1 bit a derecha o izquierda con o sin acarreo.

Si bien en el 8048 no existen instrucciones de substracción, esta operación puede ser fácilmente realizada con tres instrucciones de un solo octeto y un solo ciclo. Un operando puede ser restado del acumulador y el resultado vuelto a introducir en el mismo mediante el complemento del acumulador, sumando el valor al acumulador y nuevo complemento del acumulador.

Operaciones con registros

Se puede acceder a los registros de trabajo a través del acumulador tal como se ha explicado anteriormente, o bien pueden ser cargados de forma inmediata a partir de contenidos almacenados en la memoria de programa. Además, pueden ser incrementados o decrementados o usados en lazos contadores usando el decremento y salto si la instrucción no es 0, tal como se explicará más adelante.

Toda la memoria de datos, incluyendo los registros de trabajo, es accesible mediante instrucciones con direccionamiento indirecto a través de R_0 y R_1.

Indicadores (Flags)

En el 8048 existen cuatro indicadores accesibles para el usuario: acarreo, acarreo auxiliar, F0 y F1. El acarreo indica que ha habido rebasa-

miento de capacidad en el acumulador, mientras que el acarreo auxiliar indica rebasamiento de capacidad entre dígitos en código BCD y es utilizado en operaciones de ajuste decimal. Ambos, acarreo y acarreo auxiliar, son accesibles a través de la palabra de estado siendo almacenados en la pila durante la ejecución de las subrutinas. F0 y F1 son indicadores de aplicación general utilizables en función de las necesidades del programa. Ambos indicadores pueden ser borrados o complementados y examinados mediante instrucciones de salto condicional. F0 es accesible también a través de la palabra de estado y puede ser almacenado en un registro junto con los indicadores de acarreo.

Instrucciones de bifurcación

La instrucción de salto incondicional es de dos octetos y permite saltos a cualquier posición del primer banco de 2K palabras de la memoria de programa. Saltos al segundo banco de 2K de la memoria (pueden ser directamente direccionables hasta 4K), pueden realizarse primero mediante la ejecución de una instrucción de selección de banco de memoria y después mediante la ejecución de la instrucción de salto propiamente dicha. El límite de 2K sólo puede ser superado mediante instrucciones de salto o llamada a subrutina, es decir, el cambio de banco no se realiza hasta haber ejecutado una instrucción de salto. Una vez el banco de memoria ha sido seleccionado, los siguientes saltos serán a dicho banco hasta que se ejecute otra instrucción de selección de banco de memoria. Una subrutina que se encuentre en el banco opuesto puede ser accedida mediante una instrucción de selección de banco de memoria seguida de otra de llamada a subrutina. Una vez completada la ejecución de la subrutina se retornará automáticamente al banco de origen; no obstante, en caso de no haber seleccionado de nuevo el banco de origen, la siguiente instrucción de salto que se encuentre será de nuevo transferida al banco opuesto.

Los saltos condicionales pueden examinar las siguientes entradas y estados internos:

- Terminal de entrada T0.
- Terminal de entrada T1.
- Terminal de entrada INT.
- Acumulador a cero.
- Cualquier bit del acumulador.
- Indicador de acarreo.
- Indicador F0.
- Indicador F1.

Los saltos condicionales permiten una bifurcación a cualquier posición de una página de memoria (256 octetos) en ejecución. Las condiciones

examinadas son valores instantáneos en el momento de ejecutarse un salto condicional. Por ejemplo, la instrucción de salto, si el acumulador está a cero, examina el acumulador ella misma, no un indicador de cero intermedio.

La instrucción de decremento de un registro y salto si no es cero, combina una instrucción de decremento y otra de bifurcación para crear una instrucción muy útil en la implementación de lazos contadores. Esta instrucción puede designar cualquiera de los 8 registros de trabajo y efectuar una bifurcación a cualquier dirección de la página que se esté ejecutando.

La instrucción de salto indirecto de un solo octeto permite acceder a cualquier posición de memoria basándose en el contenido del acumulador. El contenido del acumulador apunta a la posición de la memoria de programa que contiene la dirección de salto. La dirección de salto, de 8 bits, está referido a la página en ejecución. Esta instrucción podría ser usada, por ejemplo, para acceder a cualquiera de las diferentes rutinas basadas en caracteres ASCII, los cuales son cargados en el acumulador. De esta forma las entradas de teclas ASCII pueden ser usadas para iniciar diferentes rutinas.

Subrutinas

El salto a subrutina se ejecuta mediante la instrucción CALL. Dicha instrucción puede generar saltos incondicionales a cualquier posición de un banco de memoria de 2K y de la misma forma saltar por encima del límite de 2K. Dos instrucciones de retorno distintas determinan si se restablece o no el contenido de la PSW (cuatro bits más significativos) al retornar de una subrutina.

La instrucción de retorno y restablecimiento del contenido de la PSW también indica el final de una subrutina de interrupción, si existe alguna en proceso.

Instrucciones del temporizador

El temporizador/contador de 8 bits puede ser cargado o leído a través del acumulador, bien cuando está parado, bien cuando está contando. El contador puede ser activado mediante un reloj interno, realizando las funciones de un temporizador, o bien realizar las funciones de contador de impulsos o temporizador a partir de un reloj externo aplicado a la patilla de entrada T1. La instrucción ejecutada determina qué fuente de

impulsos se utiliza. Una sola instrucción detiene el contador, independientemente de que esté operando con una fuente de reloj interna o externa. Además, dos instrucciones distintas permiten inhibir o desinhibir el impulso de interrupción del temporizador.

Instrucciones de control

Dos instrucciones permiten que la fuente de interrupciones externa pueda ser inhibida o desinhibida. Inicialmente, las interrupciones se encuentran desinhibidas y son automáticamente inhibidas durante la ejecución de una subrutina de interrupción, siendo otra vez desinhibidas al finalizar la misma.

Existen cuatro instrucciones de selección de banco de memoria, dos para designar el banco de registros de trabajo efectivo a utilizar y dos para controlar los bancos de memoria de programa. Las instrucciones de cambio de registros de trabajo permite al programador la inmediata sustitución por un segundo banco de 8 registros de trabajo del que tengan en uso en ese momento. Esto permite disponer de 16 registros de trabajo o bien puede ser usado como medio rápido para salvar el contenido de los registros cuando se produce una interrupción. El usuario tiene la opción de cambiar o no cambiar de banco cuando se produce una interrupción. No obstante, si los bancos son cambiados, el banco de origen será atomáticamente restituido durante la ejecución de una instrucción de re retorno y restablecimiento del estado de la PSW al final de una subrutina de interrupción.

Una instrucción especial desinhibe un reloj interno de frecuencia igual a un tercio de la del cristal y que tiene su salida por la patilla T0. Este reloj puede ser usado como aplicación general de un reloj en un sistema. Esta instrucción será utilizada solamente al inicializar el sistema, ya que la salida de reloj puede ser inhibida únicamente mediante la aplicación de la puesta a cero del sistema.

Instrucciones de Entrada/Salida

Los registros de E/S 1 y 2 son registros estáticos de E/S de 8 bits los cuales pueden ser cargados hacia o desde el acumulador. Las salidas son almacenadas estáticamente, pero las entradas no son almacenadas y se hace necesario leerlas mientras están presentes. Además, datos inmediatos de la memoria de programa pueden ser sometidos a funciones AND u OR directamente con los registros de E/S 1 y 2, quedando el resultado en los registros de E/S. Esto permite realizar máscaras almacenadas en la memo-

ria de programa, para seleccionar la puesta 1 o a 0 de bits individuales de los registros de E/S. Los registros de E/S 1 y 2 están estructurados de forma que permitan la entrada de información por una patilla seleccionada, siempre que antes escribamos por programa un 1 en dicha patilla.

Un registro de E/S de 8 bits, llamado BUS, puede ser accedido también a través del acumulador y puede tener salidas con almacenamiento temporal estático. En él se pueden realizar funciones AND y OR entre datos inmediatos de la memoria de programa directamente con sus salidas; no obstante, este proceso es distinto al de los registros de E/S 1 y 2, ya que se necesitan tratar los 8 bits del BUS a la vez en todo momento, sean entradas o salidas. Además de ser un registro de E/S estático, el BUS puede ser usado como un registro de E/S bidireccional síncrono usando las instrucciones de transferencia externa que permiten acceder a la memoria de datos externa. Cuando estas instrucciones son ejecutadas, se genera el correspondiente impulso de lectura o escritura y se hace válido el dato únicamente durante ese tiempo. Mientras no hay transferencia de datos al BUS, éste se encuentra en estado de alta impedancia.

Los tres registros de E/S pueden ser expandidos externamente mediante la mitad del registro de E/S 2 constituido en la barra expansora de cuatro bits. Los dispositivos expansores de E/S en esta barra constan de cuatro registros de E/S de cuatro bits, direccionables como registros de E/S del 4 al 7. Estos registros de E/S tienen sus propias instrucciones AND y OR del expansor combinan el contenido del acumulador con el registro de E/S internos lo hacen directamente con un dato inmediato.

Dispositivos de E/S pueden también ser añadidos externamente usando el registro BUS de E/S como barra de expansión. En este caso los registros de E/S son direccionados como si se tratara de la memoria de datos externa, encontrándose direccionado en el espacio de direcciones de la memoria de datos externa por los registros punteros R0 y R1.

Instrucciones del 8048

Símbolos y abreviaciones utilizadas

A	Acumulador
AC	Acarreo auxiliar
addr	Dirección de memoria de programa de 12 bits
Bb	Designador del bit (b = 0 – 7)
BS	Conmutador de banco
BUS	Registro de E/S de barra
C	Acarreo
CLK	Reloj
CNT	Contador de sucesos

(Continúa)

(Continuación)

D	Nemónico de un dígito de 4 bits (Nibble)
data	Dato de 8 bits
DBF	Biestable de banco de memoria
F0, F1	Indicador 0, indicador 1
I	Interrupción
P	Nemónico de operación
PC	Contador de programa
Pp	Designador de registro de E/S (p = 1,2 ó 4-7)
PSW	Palabra de estado de programa
Rr	Designador de registro
SP	Puntero de pila
T	Temporizador
TF	Indicador de temporizador
T0, T1	Test 0, test 1
X	Nemónico para RAM externa
#	Prefijo de dato inmediato
@	Prefijo de dirección indirecta
$	Valor normal del contador de programa
(X)	Contenido de X
((X))	Contenido de una posición direccionada por X
←	Reemplazado por

TABLA 2. TABLA DE INSTRUCCIONES

Nemónico	Descripción	Código		Código hexadecimal	N.º octetos	N.º ciclos
Acumulador						
ADD A,R	Suma del registro R a A	0 1 1 0	1 r r r	–	1	1
ADD A, @ R	Suma del dato direccionado por R a A	0 1 1 0	0 0 0 r	–	1	1
ADD A, # data	Suma inmediata de A	0 0 0 0	0 0 1 1	0 3	2	2
ADDC A,R	Suma de R a A con acarreo	0 1 1 1	1 r r r	–	1	1
ADDC A @ R	Suma del dato direccionado por R a A con acarreo	0 1 1 1	0 0 0 r	–	1	1
ADDC A, # data	Suma inmediata de A con acarreo	0 0 0 1	0 0 1 1	1 3	2	2
ANL A,R	AND entre el registro R y A	0 1 0 1	1 r r r	–	1	1
ANL A, @ R	and entre el dato direccionado por R y A	0 1 0 1	0 0 0 r	–	1	1
ANL A, # data	AND inmediata con A	0 1 0 1	0 0 1 1	5 3	2	2
ORL A,R	OR entre el registro R y A	0 1 0 0	1 r r r	–	1	1
ORL A, @ R	OR entre el dato direccionado por R y A	0 1 0 0	0 0 0 r	–	1	1
ORL A, # data	OR inmediato con A	0 1 0 0	0 0 1 1	4 3	2	2

TABLA 2. TABLA DE INSTRUCCIONES (Continuación)

Nemónico	Descripción	Código		Código hexadecimal	N.º octetos	N.º ciclos
XRL A,R	OR exclusivo entre el registro R y A	1 1 0 1	1 r r r	–	1	1
XRL A, @ R	OR exclusivo entre el dato direccionado por R y A	1 1 0 1	0 0 0 r	–	1	1
XRL A, # data	OR exclusivo inmediato con A	1 1 0 1	0 0 1 1	D3	2	2
INC A	Incrementar A	0 0 0 1	0 1 1 1	1 7	1	1
DEC A	Decrementar A	0 0 0 0	0 1 1 1	0 7	1	1
CLR A	Poner a cero A	0 0 1 0	0 1 1 1	2 7	1	1
CPL A	Complementar el contenido de A	0 0 1 1	0 1 1 1	3 7	1	1
DA A	Ajuste decimal de A	0 1 0 1	0 1 1 1	5 7	1	1
SWAP A	Los bits 0 a 3 son cambiados por los bits 4 a 7	0 1 0 0	0 1 1 1	4 7	1	1
RL A	Rotación de A a la izquierda	1 1 1 0	0 1 1 1	E 7	1	1
RLC A	Rotación de A a la izquierda a través del acarreo	1 1 1 1	0 1 1 1	E 7	1	1
RR A	Rotación de A a la derecha	0 1 1 1	0 1 1 1	E 7	1	1
RRC A	Rotación de A a la derecha a través del acarreo	0 1 1 0	0 1 1 1	6 7	1	1
Entrada-Salida						
IN A,P	Entrar el contenido de un registro de E/S (1 ó 2) y transferirlo a A	0 0 0 0	1 0 p p	–	1	2
OUTL P, A	Sacar el contenido de A y transferirlo a un registro de E/S (1 ó 2)	0 0 1 1	1 0 p p	–	1	2
ANL P, # data	AND inmediata con un registro de E/S (1 ó 2)	1 0 0 1	1 0 p p	–	2	2
ORL, P, # data	OR inmediato con un registro de E/S (1 ó 2)	1 0 0 0	1 0 p p	–	2	2
INS A, BUS	Entrar el contenido de la barra y transferirlo a A	0 0 0 0	1 0 0 0	0 8	1	2
OUTL BUS, A	Sacar el contenido de A y transferirlo a la barra	0 0 0 0	0 0 1 0	0 2	1	2
ANL BUS, # data	AND inmediata con la barra	1 0 0 1	1 0 0 0	9 8	2	2
ORL BUS, # data	OR inmediata con la barra	1 0 0 0	1 0 0 0	8 8	2	2
MOVD A,P	Entrar el contenido de un registro de E/S externo (4 – 7) y transferirlo a A	0 0 0 0	1 1 p p	–	1	2
MOVD P,A	Sacar el contenido de A y transferirlo a un registro de E/S externo (4 – 7)	0 0 1 1	1 1 p p	–	1	2

TABLA 2. TABLA DE INSTRUCCIONES (Continuación)

Nemónico	Descripción	Código	Código hexadecimal	N.º octetos	N.º ciclos
ANLD P,A	AND entre un registro de E/S externo y A	1 0 0 1	1 1 p p	–	1	2
ORLD P,A	OR entre un registro de E/S externo y A	1 0 0 0	1 1 p p	–	1	2
Registros						
INC R	Incrementar el registro R	0 0 0 1	1 r r r	–	1	1
INC @ R	Incrementar el dato direccionado por R	0 0 0 1	0 0 0 r	–	1	1
DEC R	Decrementar el registro R	1 1 0 0	1 r r r	–	1	1
Saltos						
JMP addr	Salto incondicional a la dirección especificada	$a_{10}a_9a_8$0	0 1 0 0	–	2	2
JMPP @ A	Salto indirecto (direccionado por A)	1 0 1 1	0 0 1 1	B 3	1	2
DJNZ R,addr	Decremento y examen de R. Salta si (R) $\neq$ 0	1 1 1 0	1 r r r	–	2	2
JC addr	Salto si el acarreo es 1	1 1 1 1	0 1 1 0	F 6	2	2
INC addr	Salto si el acarreo es 0	1 1 1 0	0 1 1 0	E 6	2	2
JZ addr	Salto si (A) = 0	1 1 0 0	0 1 1 0	C 6	2	2
JNZ addr	Salto si (A) $\neq$ 0	1 0 0 1	0 1 1 0	9 6	2	2
JTO addr	Salto si TO es 1	0 0 1 1	0 1 1 0	3 6	2	2
JNTO addr	Salto si TO es 0	0 0 1 0	0 1 1 0	2 6	2	2
JT1 addr	Salto si T1 es 1	0 1 0 1	0 1 1 0	5 6	2	2
JNT1 addr	Salto si T1 es 0	0 1 0 0	0 1 1 0	4 6	2	2
JFO addr	Salto si el indicador 0 (F0) es 1	0 1 1 1	0 1 1 0	B 6	2	2
JF1 addr	Salto si el indicador 1 (F1) es 1	0 1 1 1	0 1 1 0	7 6	2	2
JTF addr	Salto si el indicador del temporizador/contador (FT) es 1)	0 0 0 1	0 1 1 0	1 6	2	2
JN1 addr	Salto si la entrada de interrupción externa (INT) es 0	1 0 0 0	0 1 1 0	8 6	2	2
JBb addr	Salto si el bit b (1 entre 8) de A es 1	$b_2b_1b_0$1	0 0 1 0	–	2	2
Subrutinas						
CALL	Salto a subrutina	$a_{10}a_9a_8$1	0 1 0 0	–	2	2
RET	Retorno de subrutina	1 0 0 0	0 0 1 1	8 3	1	2

TABLA 2. TABLA DE INSTRUCCIONES (Continuación)

Nemónico	Descripción	Código	Código hexadecimal	N.º octetos	N.º ciclos
RETR	Retorno de subrutina y restablecimiento del estado inicial	1001 0011	93	1	2
Indicadores					
CLR C	Puesta a cero de acarreo	1001 0111	97	1	1
CPL C	Complementar el acarreo	1010 0111	A7	1	1
CLR F0	Puesta a cero de F0	1000 0101	85	1	1
CPL F0	Complementar F0	1001 0101	95	1	1
CLR F1	Puesta a cero de F1	1010 0101	A5	1	1
CPL F1	Complementar F1	1011 0101	B5	1	1
Transferencia de datos					
MOV A,R	Transferencia de F a A	1111 1rrr	—	1	1
MOV A, @R	Transferencia de un dato direccionado por R a A	1111 000r	—	1	1
MOV A, #data	Carga inmediata de A	0010 0011	23	2	2
MOV R,A	Transferencia de A a R	1010 1rrr	—	1	1
MOV @R,A	Transferencia de A a una posición de memoria direccionada por R	1010 000r	—	1	1
MOV R, #data	Carga inmediata de R	1011 1rrr	—	2	2
MOV @R, #data	Transferencia inmediata a una posición de memoria direccionada por R	1011 000r	—	2	2
MOV A,PSW	Transferencia de la palabra de estado (PSW) a A	1100 0111	C7	1	1
MOV PSW,A	Transferencia de A a la palabra de estado (PSW)	1101 0111	D7	1	1
XCH A,R	Cambio de A y R	0010 1rrr	—	1	1
XCH A, @R	Cambio de A y de un dato direccionado por R	0010 000r	—	1	1
XCHD A, @R	Cambio de A y 4 bits de un dato direccionado por R	0011 000r	—	1	1
MOVX A, @R	Transferencia de un dato direccionado por R a A	1000 000r	—	1	2
MOVX @R,A	Transferencia de A en la posición de memoria direccionada por R	1001 000r	—	1	2
MOVP A, @A	Transferencia de un dato direccionado por A a A	1010 0011	A3	1	2
MOVP3 A, @A	Transferencia de un dato direccionado por A a A	1110 0011	E3	1	2

TABLA 2. TABLA DE INSTRUCCIONES (Continuación)

Nemónico	Descripción	Código		Código hexadecimal	N.º octetos	N.º ciclos
Temporizador/contador						
MOV A,T	Transferencia del temporizador/contador a A	0 1 0 0	0 0 1 0	4 2	1	1
MOV T,A	Transferencia de A al temporizador/contador	0 1 1 0	0 0 1 0	6 2	1	1
STRT T	Arranque del temporizador	0 1 0 1	0 1 0 1	5 5	1	1
STRT CNT	Arranque del contador de impulsos	0 1 0 0	0 1 0 1	4 5	1	1
STOP TCNT	Para del temporizador o contador	0 1 1 0	0 1 0 1	6 5	1	1
EN TCNT1	Desinhibición de las interrupciones del temporizador/contador	0 0 1 0	0 1 0 1	2 5	1	1
DIS TCNT1	Inhibición de las interrupciones del temporizador/contador	0 0 1 1	0 1 0 1	3 5	1	1
Control						
EN I	Desinhibición de las interrupciones externas	0 0 0 0	0 1 0 1	0 5	1	1
DIS I	Inhibición de las interrupciones externas	0 0 0 1	0 1 0 1	1 5	1	1
SEL RB0	Selección del banco de registros 0	1 1 0 0	0 1 0 1	C 5	1	1
SEL RB1	Selección del banco de registros 1	1 1 0 1	0 1 0 1	D 5	1	1
SEL MB0	Selección del banco de memoria 0	1 1 1 0	0 1 0 1	E 5	1	1
SEL MB1	Selección del banco de memoria 1	1 1 1 1	0 1 0 1	F 5	1	1
ENTO CLK	Salida del reloj interno por TO	0 1 1 1	0 1 0 1	7 5	1	1
NOP	No operación	0 0 0 0	0 0 0 0	0 0	1	1

CAPITULO 8

Ejemplos y proyectos de carácter didáctico de sistemas con microprocesador

INTRODUCCION

Este capítulo va destinado a familiarizar al lector con la realización de programas y a la implementación de sistemas básicos con microprocesadores. Sólo resolviendo programas, probándolos, depurándolos, implementando el hardware que forma el sistema y realizando la puesta en marcha en condiciones reales se puede conseguir una práctica inicial, que permita acometer problemas más complejos.

Para grabar fácilmente los programas y probarlos, aparte de equipos de desarrollo completos y sofisticados, existen kits más sencillos y económicamente accesibles, que pueden emplearse en una primera fase de estudio y experimentación. En la figura 8-1 a) se ofrece un aspecto de un equipo

Fig. 8-1 a).– Foto del sistema de desarrollo para los microprocesadores R 6500. (Cortesía de Rockwell).

de desarrollo profesional para diseño de sistemas industriales, basados en los microprocesadores de la familia R 6500.

La figura 8-1 b) muestra dos módulos microcomputadores básicos, el AIM-65 y el KIM-1, con los que se puede probar e implementar programas y proyectos limitados.

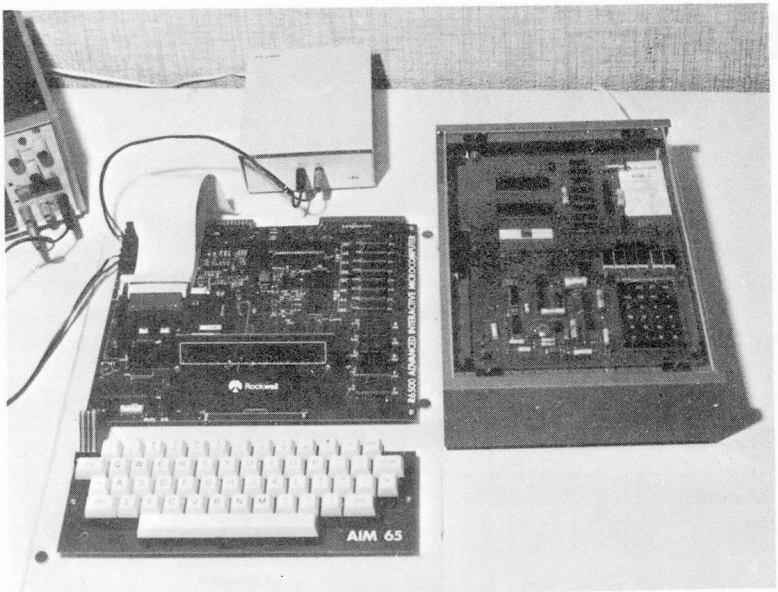


Fig. 8-1 b).–Foto de 2 microcomputadores: el AIM-65 a la izquierda y el KIM-1 a la derecha, para trabajo con la familia R 6500 (Cortesía de ANCART).

En la figura 8-2 se presenta un diagrama gráfico de las operaciones que se deben realizar para llevar a cabo un proyecto basado en microprocesadores, utilizando el sistema de desarrollo, en este ejemplo el System 65 mostrado en la figura 8-1 a), el cual tiene capacidad de grabar memorias PROM.

La experiencia necesaria para acometer nuevos diseños no se consigue únicamente comprendiendo el desarrollo de algunos programas ya confeccionados, es necesario, cargarlos en memoria, ejecutarlos y analizar los resultados obtenidos.

Los ejemplos que se ofrecen comienzan con ejemplos muy simples, que exponen y resuelven cálculos aritméticos y lógicos, precisándose únicamente para su comprobación un microcomputador cualquiera. Los siguientes ejemplos tratan de resolver aplicaciones simples, que requieren

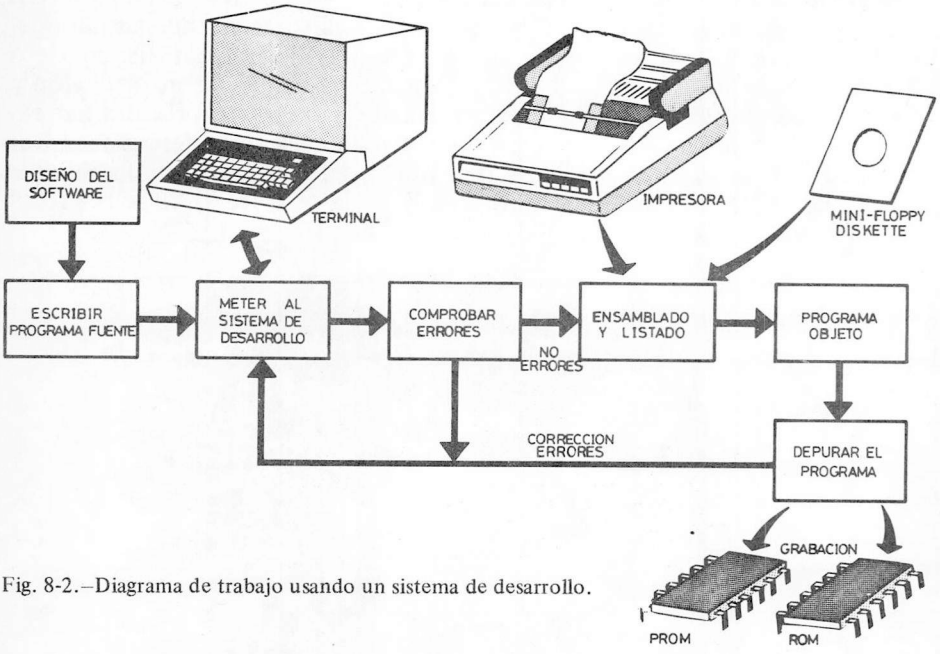


Fig. 8-2.–Diagrama de trabajo usando un sistema de desarrollo.

una conexión externa del microcomputador a los periféricos que ha de controlar (relés, contadores, leds, etc.) y que, por lo tanto, necesitan de circuitos externos auxiliares con hardware complementario.

Ideas más completas sobre el manejo del hardware, que rodea a los sistemas basados en microprocesadores, se obtendrán con la lectura de la parte de la obra destinada a la explicación de la arquitectura de varios microcomputadores, en donde quedan resueltos la mayoría de los problemas que surgen en la práctica y que está contenida en otro libro.

Aunque es norma general confeccionar el diagrama de flujo antes de realizar el programa de instrucciones, en los ejemplos muy sencillos se ha prescindido de este paso. Los programas, que resuelven los ejemplos de este capítulo "con fines didácticos", se han escrito no sólo con nemónicos, siguiendo el lenguaje Ensamblador, sino también en código OP (hexadecimal) con indicación de las posiciones de memoria que ocupa cada instrucción, operando y con comentarios aclaratorios.

Para ofrecer una clara visión de la similitud que existe en trabajar con cualquier tipo de microprocesador, cada ejemplo se ha tratado de resolver con dos microprocesadores diferentes, el Motorola 6800 y el corres-

pondiente a la familia R 6500. Para resolver estos ejercicios puede usarse, en el caso del Motorola, el kit que comercializa la misma casa, modelo MEK 6800 D2. Consta de 2 placas, una de ellas con un display de 6 dígitos con su interface, un teclado para introducción de información y un interface para cassette de audio, que actúa como memoria auxiliar externa; en la otra placa reside el sistema básico a base del microprocesador 6800. También para este mismo microprocesador es muy recomendable con fines didácticos el microcomputador de la casa Heathkit, cuya fotografía se muestra en la figura 8-3.

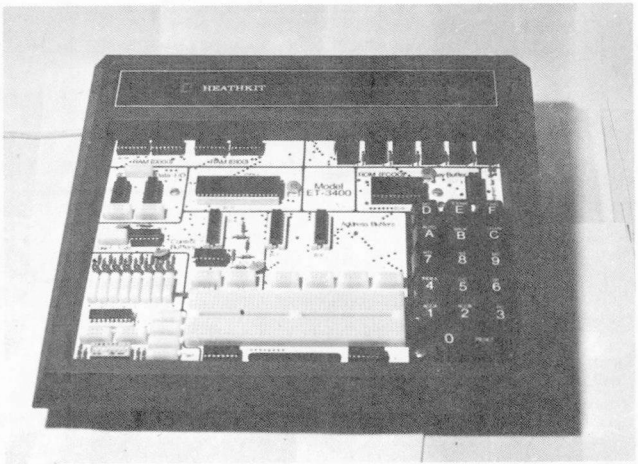
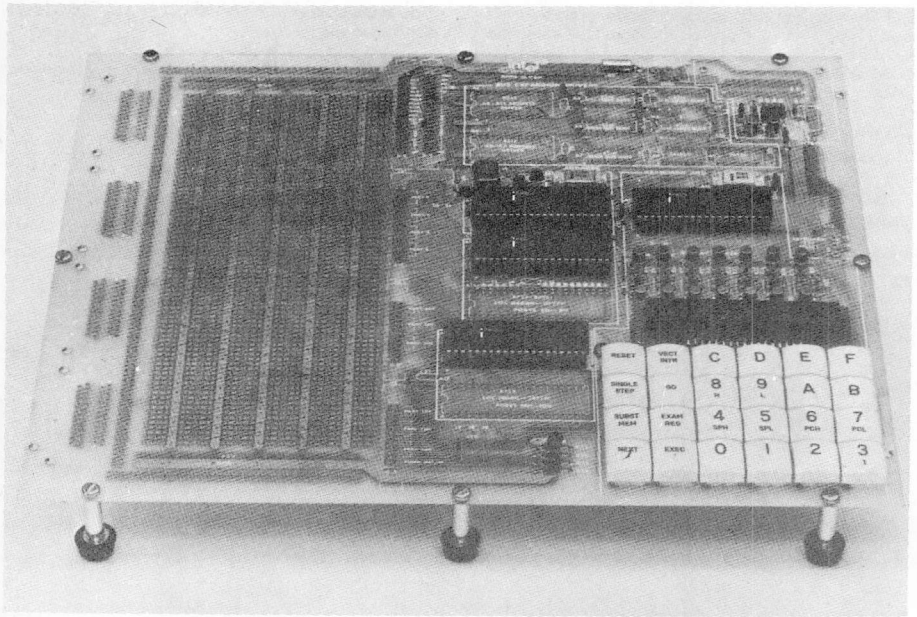


Fig. 8-3.–Fotografía del microcomputador Heathkit. (Cortesía de Academia ANCART).

Para desarrollar los ejercicios con los microprocesadores R 6500 pueden usarse los microcomputadores AIM-65 y KIM-1 que aparecen fotografiados en la figura 8-1 b).

Todos los fabricantes de microprocesadores ofrecen a sus usuarios mi crocomputadores similares a los comentados, bien directamente o bien a través de segundas casas. Así Intel ofrece el kit SDK-85, sistema de desarrollo para proyectos con el microprocesador 8085, versión mejorada del 8080. (Ver figura 8-4.)

Aunque todos los microcomputadores emplean el lenguaje del microprocesador que lo gobierna, cada uno dispone de una serie de comandos y ayudas a la programación, que se derivan del programa Monitor residente en su memoria ROM. Esto supone que el lector puede alcanzar cierto dominio de la programación y del hardware auxiliar, siguiendo los



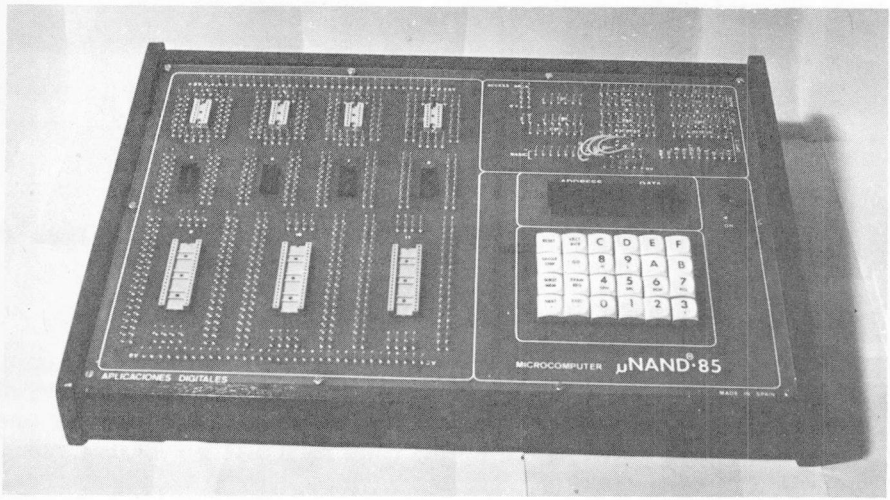


Fig. 8-4.–En la parte superior, el kit SDK-85 Intel montado. Abajo el mismo kit montado sobre un panel por Aplicaciones Digitales con fines de experimentación y didácticos.

ejercicios de este capítulo con cualquier microcomputador que emplee los mismos microprocesadores, pero cada uno de estos aparatos dispondrá de diferente programa Monitor, que ofrecerá un conjunto de operaciones posibles que hay que conocer con detalle.

La fotografía 8-4 bis muestra la maleta, que contiene el microcomputador de Hewlett Packard 5063A para entrenamiento sobre el 8085 de INTEL, cuya arquitectura se describe con detalle en otro tomo de esta obra, junto con proyectos y aplicaciones basadas en microprocesadores INTEL.

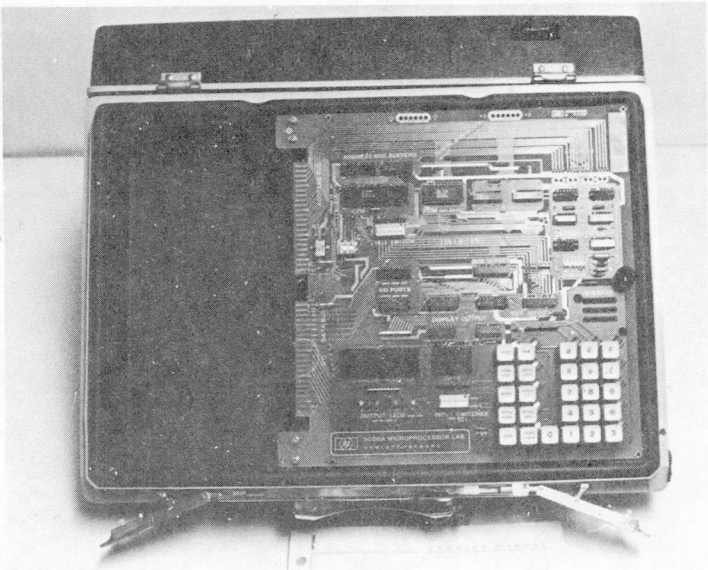


Fig. 8-4 Bis.–Fotografía de microcomputador de Hewlett Packard basado en el INTEL 8085. (Cortesía del Dpto. Arquitectura de Ordenadores de la Universidad de Deusto.)

Para la mejor comprensión de los problemas y diseños, que se pasan a describir en este capítulo, se remite al lector al estudio de los microcomputadores con los que se puede llevar a cabo que, como se ha mencionado anteriormente, se presentan en un volumen posterior de esta misma obra: MICROPROCESADORES (Fundamentos, Diseño y Aplicaciones).

1.º EJERCICIO (R 6500)

Se trata de realizar la suma de dos datos X e Y, almacenados en dos posiciones de memoria, la 0F y la 10, debiéndose cargar el resultado de la operación en la posición de memoria número 11. Todas las posiciones de memoria se refieren a la página 0, por lo que el primer byte de la dirección será 00.

Por la sencillez del ejercicio no se confecciona el ordinograma y se ofrece directamente su solución en el siguiente cuadro:

Comentario	Nemónico	Operando	OP	Dirección
Flag D = 0 (Forma binaria)	CLD		D8	0000
A ← (0F)	LDA	0F	A5	0001
			0F	0002
C ← 0	CLC		18	0003
A + (10) → A,C	ADC	10	65	0004
			10	0005
A → (11)	STA	11	85	0006
			11	0007
Paro	BRK		00	0008
1.º sumando				000F
2.º sumando				0010
Resultado				0011

La primera instrucción del programa (CLD) pasa a 0 el flag D, lo que sirve para hacer que la ALU realice las operaciones aritméticas en forma binaria. Si D = 1, la ALU trabaja en forma decimal, es decir, en BCD.

La 2.ª instrucción carga en el Acumulador el 1.º sumando, situado en la posición de memoria 000F, que previamente a la ejecución del programa habrá de cargarse con el valor que se desee asignar al 1.º sumando.

Posteriormente se borra, ó se pone a 0, el flag de acarreo (carry) y se suma al Acumulador el 2.º sumando, situado en la posición 0010. Por último, el resultado de la suma, que queda depositado en el Acumulador, se almacena en la posición 0011 mediante la instrucción STA, 11.

Recuérdese que la instrucción BRK, que ocupa la posición 0008 del programa origina una secuencia de interrupción del tipo $\overline{IRQ}$, almacenando en el Stack al PC y al registro de estado. El PC pasa a contener el valor del vector de interrupción, asignado en general a las posiciones de memoria FFFE y FFFF. Caso de trabajar con el microcomputador KIM-1, es

preciso, antes de ejecutar el programa, cargar el vector de interrupción de la siguiente forma para que actúe la instrucción BRK como "paro":

17 FE ← 00
17 FF ← 1C

RESOLUCION DEL MISMO EJERCICIO CON INSTRUCCIONES DEL MOTOROLA 6800

Comentario	Nemónico	Operando	OP	Dirección
C ← 0	CLC		0C	0000
A ← (0F)	LDAA	0F	96	0001
			0F	0002
A + (10) → A,C	ADDA	10	9B	0003
			10	0004
A ← (11)	STAA	11	97	0005
			11	0006
Paro	HALT		3E	0007
1.º sumando				000F
2.º sumando				0010
Resultado				0011

Dado que el tipo de instrucciones utilizadas ahora es el mismo que cuando se empleó el microprocesador de la familia R 6500, prácticamente solo varían los códigos OP, elegidos en este caso para el microprocesador Motorola 6800, sólo se ha expuesto el programa correspondiente a este ejercicio.

2.º EJERCICIO (R 6500)

En el ejemplo anterior se efectuaba una suma sobre 2 números variables, que se cargaban en una determinada dirección de memoria. Ahora se pretende realizar la misma operación, pero proporcionando los datos a sumar directamente en el programa. Al elegir el código de las instrucciones habrá que escoger el que corresponda al direccionamiento inmediato, en lugar del direccionamiento por página cero.

Obsérvese que los operandos inmediatos, que siguen a las instrucciones, se distinguen por llevar antepuesto el signo ≠ y en este ejemplo son los valores de los sumandos: 10 y 6.

Comentario	Nemónico	Operando	OP	Dirección
D ← 1	SED		F8	0000
A ← 10	LDA	#10	A9	0001
			10	0002
C ← 0	CLC		18	0003
A + 6 → A,C	ADC	#06	69	0004
			06	0005
A → (11)	STA	11	85	0006
			11	0007
Paro	BRK		00	0008
Resultado				0011

Como quiera que el programa comienza con la instrucción SED, el resultado se ofrece en decimal, por lo tanto, la suma de 10 + 6 = 16. Los códigos OP de las instrucciones cambian respecto al ejercicio anterior por variar el modo de direccionamiento.

RESOLUCION DEL 2.º EJERCICIO CON INSTRUCCIONES DEL MOTOROLA 6800

Comentario	Nemónico	Operando	OP	Dirección
C ← 0	CLC		0C	0000
A ← 10	LDAA	#10	86	0001
			10	0002
A + 6 → A,C	ADDA	#06	8B	0003
			06	0004
Ajuste decimal	DAA		19	0005
A → (11)	STAA	11	97	0006
			11	0007
Paro	HLT		3E	0008
Resultado				0011

3.º EJERCICIO (R 6500)

Se desean multiplicar dos números variables, contenidos en las posiciones 001D (multiplicando) y 001E (multiplicador). El producto quedará almacenado en la posición de memoria 001F.

Se comienza planteando el diagrama de flujo, teniendo en cuenta las operaciones aritméticas que es capaz de realizar el microprocesador, que sólo son la suma y la resta. Esto obliga a estructurar el ordinograma de la

multiplicación a base de la repetición de sumas, es decir, el producto será igual a sumar tantas veces el multiplicando como valor posea el multiplicador. El KIM-1, que puede ejecutar este ejercicio, se muestra en la figura 8-5 (1).

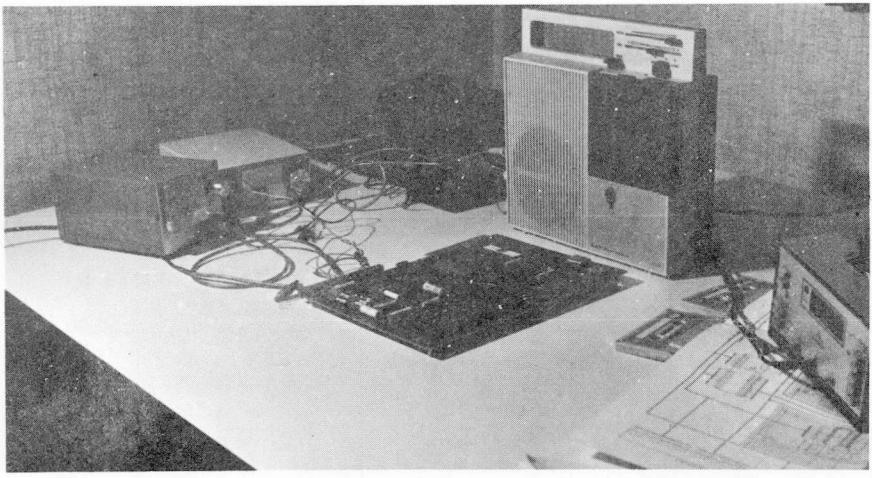


Fig. 8-5 (1).—Fotografía del KIM-1, con el que se pueda resolver los ejercicios propuestos en este capítulo, teniendo conectado la fuente de alimentación y un magnetofón para almacenamiento de programas.

La figura 8-5 (2) muestra el ordinograma de la multiplicación a base de la repetición de sumas.

El programa que resuelve el ordinograma de la figura 8-5 (2), utilizando las instrucciones que poseen los microprocesadores R 6500, es el que se muestra debajo de la mencionada figura.

Recuérdese la observación que se hizo en el 1.º ejercicio respecto al funcionamiento de la instrucción BRK, caso de emplearse el microcomputador KIM-1 para la comprobación del presente programa.

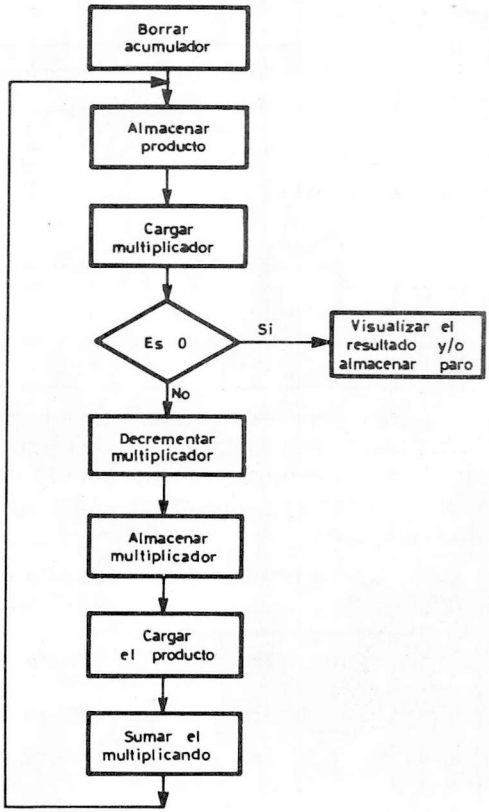


Fig. 8-5 (2).—Diagrama de flujo para resolver una multiplicación.

Comentario	Nemónico	Operando	OP	Dirección
A ← 0	LDA	≠00	A9	0000
			00	0001
A → (1F)	STA	1F	85	0002
			1F	0003
A ← (1E)	LDA	1E	A5	0004
			1E	0005
Si A = 0, salta 10 pos.	BEQ	+10	F0	0006
			0A	0007
DECREMENTAR (1E)	DEC	1E	C6	0008
			1E	0009
A ← (1F)	LDA	1F	A5	000A
			1F	000B

(Continúa)

(Continuación)

Comentario	Nemónico	Operando	OP	Dirección
C ← 0	CLC		18	000C
A + (1D) + C → A	ADC	1D	65	000D
			1D	000E
Saltar a posición 0002	JMP	0002	4C	000F
			02	0010
			00	0011
Paro	BRK		00	0012
Multiplicando				001D
Multiplicando				001E
Producto				001F

Al comprobar los resultados obtenidos con este programa, debe tenerse en cuenta que la expresión resultante puede venir dada en código binario o en el decimal, según el estado del flag D. Puede repetirse este programa añadiendo una instrucción, que obligue a dicho flag a un estado determinado.

RESOLUCION DEL 3.º EJERCICIO CON INSTRUCCIONES DEL MOTOROLA 6800

Comentario	Nemónico	Operando	OP	Dirección
A ← 0	LDAA	#00	86	0000
			00	0001
A → (1F)	STAA	1F	97	0002
			1F	0003
A ← (1E)	LDAA	1E	96	0004
			1E	0005
Si A = 0, salto 10 posiciones	BEQ	+10	27	0006
			0A	0007
DECREMENTAR (1E)	DEC	1E	7A	0008
			1E	0009
			00	000A
A ← (1F)	LDAA	1F	96	000B
			1F	000C
C ← 0	CLC		0C	000D
A + (1D) + C → A,C	ADCA	1D	99	000E
			1D	000F
Saltar a posición 0002	BRA	0002	20	0010
			F0	0011
Paro	HLT		3E	0012
Multiplicando				001D
Multiplicado				001E
Producto				001F

Obsérvese que en la posición 0010 existe un branch incondicional, que salta "hacia atrás" 16 posiciones (desde la 0011 hasta la 0002). El número 16 en hexadecimal es 10 y hay que ponerlo en forma de complemento a 2, por lo que el offset, que acompaña a la instrucción, será:

$$1\,0_{16)} = 0001\ 0000$$

Su complemento a 1: 1110 1111

$$\underline{\qquad\qquad +1}$$

Su complemento a 2: 1111 0000

En hexadecimal el número binario 1111 000 se transforma en F0.

Cualquier ejercicio puede resolverse de diferentes formas. Así el ordinograma de la multiplicación, de la figura 8-5 (3) permite confeccionar un programa para el ejercicio de la multiplicación, que reduce en 4 bytes el ofrecido anteriormente relacionado con las instrucciones del microprocesador de la familia R 6500.

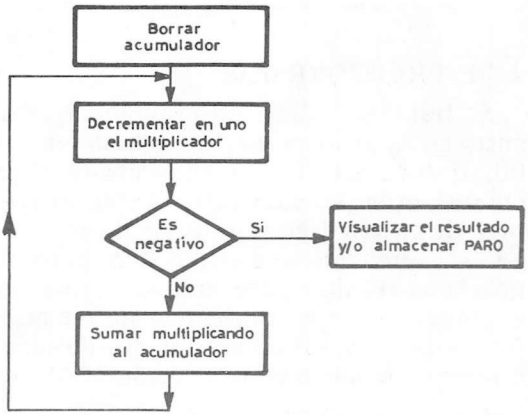


Fig. 8-5 (3).–Ordinograma simplificado para una multiplicación.

Seguidamente se ofrece el programa, que responde al ordinograma de la figura 8-5 (3).

Comentario	Nemónico	Operando	OP	Dirección
A ← 0	LDA	#00	A9	0000
			00	0001
C ← 0	CLC		18	0002

(Continúa)

(Continuación)

Comentario	Nemónico	Operando	OP	Dirección
DECREMENTA (1E)	DEC	1E	C6	0003
			1E	0004
SI MULTICDOR. (−)	BMI	05	30	0005
Salta 5 posiciones			05	0006
A + (1D) + C → A	ADC	1D	65	0007
			1D	0008
Salto a 0002	JMP	0002	4C	0009
			02	000A
			00	0008
A → (1F)	STA	1F	85	000C
			1F	000D
Paro	BRK		00	000E
Multiplicando				001D
Multiplicador				001E
Resultado				001F

4.º EJERCICIO (R 6500)

Se trata de realizar un programa que divida dos números cualesquiera entre sí. El dividendo estará situado en la posición 001F y el divisor en la 0020, debiendo quedar almacenado el cociente o resultado (sólo parte entera) en la posición 0021. Se desea visualizar el resultado sobre el display de cuatro dígitos, que, en el caso de trabajar con el microprocesador KIM-1, serán los que sirven para presentar la dirección de memoria. El programa Monitor posee una subrutina, que visualiza sobre dichos dígitos el contenido de las posiciones de memoria FA (sobre los dos dígitos de menos peso) y FB (sobre los dos de más peso). Dicha subrutina tiene su dirección de inicio en la posición 1C4F en el KIM-1.

En el caso presente, sólo se utilizarán los dos dígitos de menos peso para visuliazar el resultado, los dos de más peso deben aparecer a 0, puesto que el cociente de dividir entre sí dos posiciones de memoria de 8 bits dará como máximo resultado 8 bits ó dos dígitos hexadecimales que los representen.

Como en el caso de la multiplicación, hay que tener en cuenta que las únicas instrucciones aritméticas que admiten los microprocesadores de la familia R 6500 son las de sumar y restar. Por lo tanto, habrá que organizar la división a base de una serie de restas sucesivas.

Del dividendo habrá que restar el divisor tantas veces como se pueda, hasta llegar a una resta, que dé resultado negativo. El número de veces que hemos podido realizar dicha resta representa el valor del cociente.

En la figura 8-6 se muestra el ordinograma de la división, enpleando el procedimiento de restas sucesivas, antes comentado.

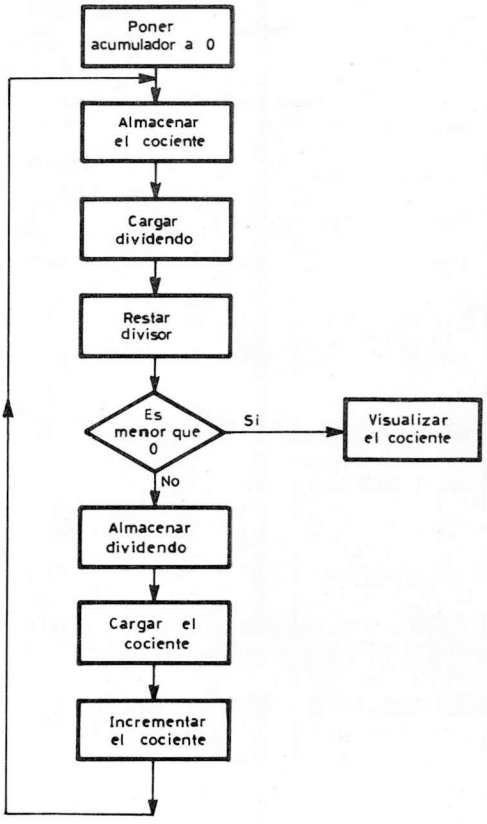


Fig. 8-6.–Ordinograma de una división para obtener la parte entera del cociente a base de repetición de restas.

El programa que resuelve el organigrama de la figura 8-6, mediante las instrucciones aceptadas por los microprocesadores R 6500, es el siguiente:

Comentario	Nemónico	Operando	OP	Dirección
A ← 0	LDA	≠00	A9	0000
			00	0001
A → (21)	STA	21	85	0002
			21	0003

(Continúa)

(Continuación)

Comentario	Nemónico	Operando	OP	Dirección
Cargar en A dividendo	LDA	1F	A5 1F	0004 0005
Se resta el divisor, pero C = 1 previamente, puesto que A − M − $\overline{C}$ → A	SEC SBC	 20	38 E5 20	0006 0007 0008
Si res. < 0 saltar +9 pos.	BMI	+9	30 09	0009 000A
Almacenar res. como divdo.	STA	1F	85 1F	000B 000C
Incrementar cociente (21)	INC	21	E6 21	000D 000E
Cargar cociente + 1	LDA	21	A5 21	000F 0010
Saltar a repetir ciclo	JMP	0002	4C 02 00	0011 0012 0013
Cargar el cociente (21)	LDA	21	A5 21	0014 0015
Almacenar coc. en FA	STA	FA	85 FA	0016 0017
A ← 0	LDA	≠00	A9 00	0018 0019
Almacenar 00 en FB	STA	FB	85 FB	001A 001B
Saltar a rutina de visualización de FA y FB, que comienza en 1C4F	JMP	1C4F	4C 4F 1C	001C 001D 001E
Dividendo Divisor Cociente o resultado				001F 0020 0021

Téngase en cuenta que el resultado de las operaciones aritméticas vendrá en código binario o en decimal, según el estado del bit o flag D.

Resolución del 4.º ejercicio con instrucciones del Motorola 6800

Siguiendo el ordinograma de la figura 8-6, el programa es el representado en el siguiente cuadro:

Comentario	Nemónico	Operando	OP	Dirección
A ← 0	LDAA	≠00	86 00	0000 0001

(Continúa)

(Continuación)

Comentario	Nemónico	Operando	OP	Dirección
ALMACENAR EN POS 21	STAA	21	97 21	0002 0003
Cargar dividendo	LDAA	1F	96 1F	0004 0005
Restar divisor	SUBA	20	90 20	0006 0007
Res. es menor que 0? salto +9	BMI	+9	2B 09	0008 0009
Almacenar nuevo divido.	STAA	1F	97 1F	000A 000B
Incrementar cociente (21)	INC	0021	7C 00 21	000C 000D 000E
Cargar cociente	LDAA	21	96 21	000F 0010
Branch incond. de −17 pos (compl 2: EF)	BRA	−17	20 EF	0011 0012
Cargar cociente	LDAA	21	96 21	0013 0014
Almacenar en pos XX	STAA	XX	97 XX	0015 0016
A ← 0	LDAA	≠00	86 00	0017 0018
Almacenar A en X′X′	STAA	X′X′	97 X′X′	0019 001A
Saltar a subrutina de visualiz. de XX y X′X′ que comienza en YYYY	JMP	YYYY	7E YY YY	001B 001C 001D
Dividendo Divisor Cociente				001F 0020 0021

Nota.: Para poder seguir el mismo flujo de instrucciones que el manejado con el microprocesador R 6500, basado su desarrollo en el KIM-1, que posee una subrutina que visualiza en 4 dígitos de un display el contenido de las posiciones FA y FB y cuyo comienzo es la posición 1C4F, se ha supuesto que con el microcomputador a base del 6800 existe una subrutina similar, que visualiza las posiciones XX y X′X′ y cuyo comienzo está en la posición YYYY. Dicha subrutina deberá programarla el propio lector, si su microcomputador no la tiene ya grabada en su programa Monitor.

En el cuadro siguiente se presenta el programa para la división, empleando instrucciones de los microprocesadores de la familia R 6500, con el que se han reducido 7 bytes con respecto al expuesto anteriormente.

Comentario	Nemotécnico	Operando	OP	Dirección
A ← 0	LDA	#00	A9	0000
			00	0001
Almacenar 0 en FB	STA	FB	85	0002
			FB	0003
Cargar en A el dividendo	LDA	1F	A5	0004
			1F	0005
Poner el carry a 1	SEC		38	0006
A ← A − M − $\overline{C}$	SBC	20	E5	0007
			20	0008
Pregunta si es fin	BMI	+5	30	0009
(cuando es negativo)			05	000A
Incrementar cociente	INC	01	E6	000B
			01	000C
Repetir el ciclo	JMP	0006	4C	000D
			06	000E
			00	000F
Cargar el cociente en A	LDA	01	A5	0010
			01	0011
Almacenar cociente en FA	STA	FA	85	0012
			FA	0013
Salto a la rutina	JMP	1C4F	4C	0014
			4F	0015
			1C	0016
Dividendo				001F
Divisor				0020
Resultado				0001

5.º EJERCICIO (R 6500)

Se pretende convertir un número expresado en código BCD a binario, o hexadecimal. El número constará de 3 dígitos de 4 bits cada uno, como corresponde a un dígito BCD. El dígito destinado a las centenas se almacenará en la posición de memoria 35, el de las decenas en la 36 y el de las unidades en la 37. El resultado final se almacenará en la posición 38, y, además, se visualizará, como en el ejercicio anterior.

Para efectuar la conversión habrá que sumar tantas veces $100_{10)}$ en código binario, que en hexadecimal es $64_{16)}$, como exprese el valor de las centenas; al resultado así obtenido habrá que sumarle tantas veces $10_{10)}$, expresado en hexadecimal (0A), como indiquen las decenas. Finalmente, al resultado así obtenido se sumará directamente el valor de las unidades, puesto que tiene el mismo significado en BCD que en binario o hexadecimal.

El anterior planteamiento queda resuelto mediante el ordinograma de la figura 8-7.

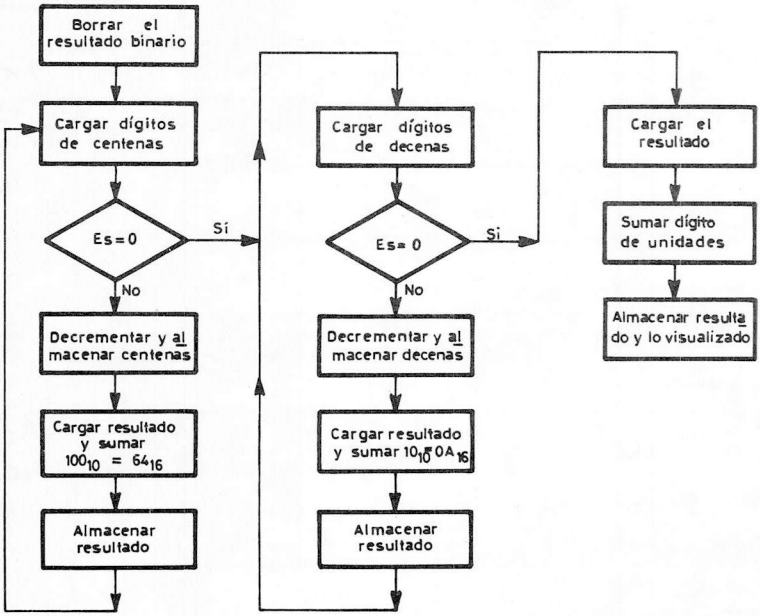


Fig. 8-7.—Ordinograma para convertir un número BCD a binario.

El programa que resuelve el ordinograma de la figura 8-7 es el siguiente:

Comentario	Nemónico	Operando	OP	Dirección
D ← 0 (forma hex.)	CLD		D8	0000
A ← 0	LDA	≠00	A9	0001
			00	0002
A → (38)	STA	38	85	0003
			38	0004
Cargar centenas	LDA	35	A5	0005
			35	0006
Si es 0 salto +12	BEQ	+12	F0	0007
			0C	0008
Decrementar centenas	DEC	35	C6	0009
			35	000A
Cargar resultado	LDA	38	A5	000B
			38	000C
C ← 0	CLC		18	000D
Sumar $100_{10} = 64_{16}$	ADC	≠64	69	000E
			64	000F

(Continúa)

(Continuación)

Comentario	Nemónico	Operando	OP	Direccion
Almacenar resultado	STA	38	85	0010
			38	0011
Repetir ciclo centena	JMP	0005	4C	0012
			05	0013
			00	0014
Cargar decenas	LDA	36	A5	0015
			36	0016
Si es 0 salto +12	BEQ	+12	F0	0017
			0C	0018
Decrementar decenas	DEC	36	C6	0019
			36	001A
Cargar resultado	LDA	38	A5	001B
			38	001C
C ← 0	CLC		18	001D
Sumar $10_{10} = 0A_{16}$	ADC	#0A	69	001E
			0A	001F
Almacenar resultado	STA	38	85	0020
			38	0021
Repetir ciclo decenas	JMP	0015	4C	0022
			15	0023
			00	0024
Cargar el resultado	LDA	38	A5	0025
			38	0026
C ← 0	CLC		18	0027
Sumar las unidades	ADC	37	65	0028
			37	0029
Almacenar el resultado	STA	38	85	002A
			38	002B
Almacenar res. en FA	STA	FA	85	002C
			FA	002D
A ← 0	LDA	#00	A9	002E
			00	002F
Cargar 00 en FB	STA	FB	85	0030
			FB	0031
Saltar a la subrutina de visualización de FA y FB	JMP	1C4F	4C	0032
			4F	0033
			1C	0034
Dígito centenas				0035
Dígito decenas				0036
Dígito unidades				0037
Resultado				0038

Resolución del 5.º ejercicio con instrucciones del Motorola 6800

A continuación, se presenta el programa de instrucciones hábiles para el microprocesador Motorola 6800, que sigue el diagrama de flujo corres-

pondiente a la figura 8-7. Se hace mención a una subrutina de visualización de las posiciones XX y X'X', cuyo inicio corresponde a la posición YYYY, que debe confeccionar el programador de acuerdo con el microcomputador de que disponga.

Comentario	Nemónico	Operando	OP	Dirección
A ← 00	LDAA	≠00	86	0000
			00	0001
Poner a 0 el resultado	STAA	38	97	0002
			38	0003
Cargar centenas	LDAA	35	96	0004
			35	0005
Si A = 0, salto +12	BEQ	+12	27	0006
			0C	0007
Decrementar centenas	DEC	0035	7A	0008
			00	0009
			35	000A
Cargar resultado	LDAA	38	96	000B
			38	000C
0 ← C	CLC		0C	000D
Sumar 100	ADDA	64	8B	000E
			64	000F
Almacenar resultado	STAA	38	97	0010
			38	0011
Saltar 16 posiciones	BRA	−16	20	0012
hacia atrás			F0	0013
Cargar decenas	LDAA	36	96	0014
			36	0015
Si A = 0, salto +12	BEQ	+12	27	0016
			0C	0017
Decrementar decenas	DEC	0036	7A	0018
			00	0019
			36	001A
Cargar resultado	LDAA	38	96	001B
			38	001C
0 ← C	CLC		0C	001D
Sumar 10	ADDA	≠0A	8B	001E
			0A	001F
Almacenar el resultado	STAA	38	97	0020
			38	0021
Salto hacia atrás de	BRA	−16	20	0022
16 posiciones			F0	0023
Cargar el resultado	LDAA	38	96	0024
			38	0025
C ← 0	CLC		0C	0026
Sumar las unidades	ADDA	37	9B	0027
			37	0028
A → (38)	STAA	38	97	0029
			38	002A

(Continúa)

(Continuación)

Comentario	Nemónico	Operando	OP	Direccion
Visualización del resultado	STAA	XX	97	002B
			XX	002C
	LDAA	00	96	002D
			00	002E
	STAA	X'X'	97	002F
			X'X'	0030
	JMP	YYYY	7E	0031
			YY	0032
			YY	0033
Dígito centenas				0035
Dígito decenas				0036
Dígito unidades				0037
Resultado				0038

La figura 8-7 (bis) muestra un ordinograma del presente ejercicio, cuyo programa correspondiente reduce 16 bytes o posiciones de memoria respecto al explicado en primer lugar para el R 6500.

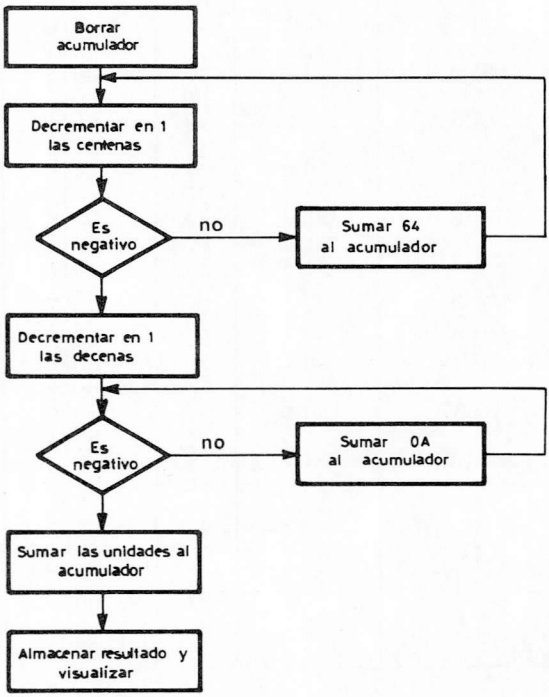


Fig. 8-7 Bis.—Ordinograma reducido para la conversión de un valor BCD a binario.

PROGRAMA REDUCIDO PARA LA CONVERSION DE UN VALOR BCD A BINARIO

Comentario	Nemónico	Operando	OP	Dirección
D ← 0 (f. Hexd.)	CLD		D8	0000
A ← 0	LDA	≠00	A9	0001
			00	0002
Decreto. centenas	DEC	35	C6	0003
			35	0004
Si negativo, salto 6 p.	BMI	06	30	0005
			06	0006
C ← 0	CLC		18	0007
A + 64	ADC	≠64	69	0008
			64	0009
Repetir ciclo centena	JMP	0003	4C	000A
			03	000B
			00	000C
Decreto. decenas	DEC	36	C6	000D
			36	000E
Si negativo, salto 6 p.	BMI	06	30	000F
			06	0010
C ← 0	CLC		18	0011
A + 0A	ADC	≠0A	69	0012
			0A	0013
Repetir ciclo decenas	JMP	000D	4C	0014
			0D	0015
			00	0016
C ← 0	CLC		18	0017
A + (37)	ADC	37	65	0018
			37	0019
Almacenar resultado	STA	38	85	001A
			38	001B
Almacenar resultado	STA	FA	85	001C
			FA	001D
A ← 0	LDA	≠00	A9	001E
			00	001F
Cargar 00 en FB	STA	FB	85	0020
			FB	0021
Salto a la subrutina de visualización de FA y FB	JMP	1C4F	4C	0022
			4F	0023
			1C	0024
Dígito centenas				0035
Dígito decenas				0036
Dígito unidades				0037
Resultado				0038

6.º EJERCICIO (R 6500)

Se trata de depositar el valor 00 en 10 posiciones contiguas de la memoria, para tratar de destacar la importancia del modo de direccionado indexado y el ahorro que, en cuanto a posiciones de memoria, supone.

El prodecimiento que se ha de seguir consistirá en cargar el valor a almacenar, 00, en el Acumulador y, a continuación, el 09 en el registro Indice X. Después, simplemente se va almacenando el Acumulador de forma indexada, es decir, que al valor u offset, que sigue a la instrucción de almacenamiento STA, se le suma el contenido del registro Indice X, que se va decrementando con cada almacenamiento.

Concretando, se desea cargar el valor 00 desde la posición 30 a la 39. El programa, que lo lleva a cabo, se expone en el siguiente cuadro.

Comentario	Nemónico	Operando	OP	Dirección
0 → A	LDA	≠00	A9	0000
			00	0001
(Reg. Indice X) ← 09	LDX	≠09	A2	0002
			09	0003
Almacenar el Ac. de modo indexado	STA	30, index	95	0004
			30	0005
Decrementar X	DEX		CA	0006
Si X es positivo saltar	BPL	−5	10	0007
5 posiciones atrás (FB)			FB	0008
Paro	BRK		00	0009

Resolución del 6.º ejercicio con instrucciones del Motorola 6800

A continuación se realiza el ejercicio anterior, que destaca la importancia del direccionamiento indexado, con instrucciones del Motorola 6800.

En el ejercicio anterior, como la instrucción DEX actúa sobre el flag N, la siguiente instrucción BPL testa el registro X, pero al no suceder esto en la instrucción correspondiente al 6800, será necesario sacar el valor del registro X a una posición, luego cargar al Acumulador dicho valor y ver si es cero.

Comentario	Nemónico	Operando	OP	Dirección
(Reg. Ind. X) ← 9	LDX	0009	CE	0000
			00	0001
			09	0002

(Continúa)

(Continuación)

Comentario	Nemónico	Operando	OP	Direccion
A ← 0	LDAA	≠00	86 00	0003 0004
Almacenar indexadamente el Acumulador	STAA	30, index.	A7 30	0005 0006
Decrementar X	DEX		09	0007
Almacenar X en posic. 40 y 41 por tener 16 bits	STX	40	DF 40	0008 0009
Cargar Ac. A con pos. 41, de menos peso	LDAA	41	96 41	000A 000B
Si el Ac. A = 0 (Z = 1), saltar 2 posiciones y paro	BEQ	+2	27 02	000C 000D
Salto incondicional de 13 posiciones hacia atrás	BRA	−13	20 F3	000E 000F
Paro	HLT	3E	3E	0010

El registro Indice X del M 6800 dispone de 16 bits y, al ser almacenado, ocupa dos posiciones de memoria, que arbitrariamente se han elegido la 40 y la 41, para posteriormente trabajar con la 41 de menos peso, en dónde se encuentra el valor significativo que se pretende testar.

7.º EJERCICIO (R 6500)

Comparar dos números, uno de ellos fijo (el 20) y el otro variable, que se almacena en la posición 17 de memoria. Si el número fijo es mayor que el variable, depositar la resta de ambos en la posición de memoria 19. Si el fijo es menor que el variable, depositar la suma de ambos en la posición 18. Finalmente, si ambos números son iguales, las 2 posiciones de memoria 18 y 19 quedarán a cero.

En la figura 8-8 se muestra el ordinograma correspondiente a este ejercicio.

Se presenta a continuación la tabla conteniendo el programa de instrucciones que sigue y resuelve el ordinograma de la figura 8-8.

Comentario	Nemónico	Operando	OP	Dirección
Se ponen a 0 las posiciones 18 y 19	LDA	≠00	A9 00	0000 0001
	STA	18	85 18	0002 0003
	STA	19	85 19	0004 0005
A ← 20	LDA	≠20	A9 20	0006 0007

(Continúa)

(Continuación)

Comentario	Nemónico	Operando	OP	Direccion
Comparar con el número	CMP	17	C5	0008
variable			17	0009
Si A es mayor, salto +6	BPL	+6	10	000A
			06	000B
C ← 0	CLC		18	000C
Sumar el fijo (20)	ADC	17	65	000D
con el variable			17	000E
Almacernar en 18	STA	18	85	000F
			18	0010
Paro	BRK		00	0011
Restar el fijo del variable	SBC	17	E5	0012
			17	0013
Almacenar en 19	STA	19	85	0014
			19	0015
Paro	BRK		00	0016
Número variable				0017
Posición con resultado (variable mayor que fijo)				0018
Posición del resultado (variable menor que fijo)				0019

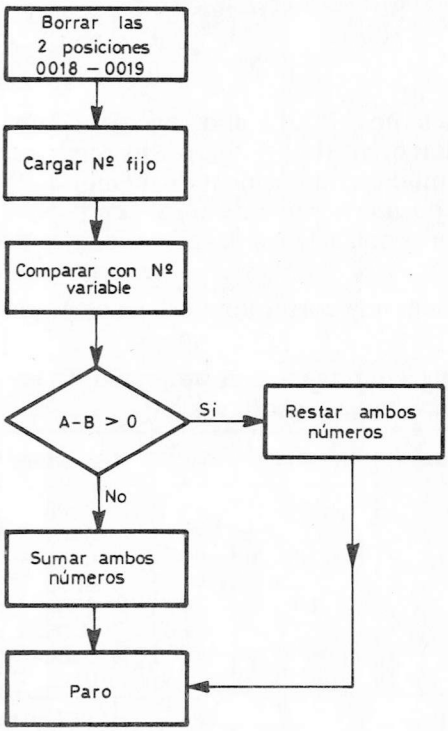


Fig. 8-8.–Ordinograma para comparar dos números según el enunciado del 7º ejercicio.

Resolución del 7.º ejercicio con instrucciones del Motorola 6800

En la siguiente tabla se presenta el programa de comparación de dos números, semejante al realizado, pero utilizando las instrucciones del microprocesador Motorola 6800.

Comentario	Nemónico	Operando	OP	Dirección
Se pone a 0 las posiciones 18 y 19	LDAA	≠00	86 00	0000 0001
	STAA	18	97 18	0002 0003
	STAA	19	97 19	0004 0005
Cargar ≠20	LDAA	≠20	86 20	0006 0007
Saltar +6 si el Ac. A es mayor que (17)	CMPA	17	91 17	0008 0009
	BPL	+6	2A 06	000A 000B
C ← 0	CLC		0C	000C
Sumar fijo + variable	ADDA	17	9B 17	000D 000E
Almacenar en 18	STAA	18	97 18	000F 0010
Paro	HLT		3E	0011
Restar fijo − variable	SUBA	17	90 17	0012 0013
Almacenar en 19	STAA	19	97 19	0014 0015
Paro	HLT		3E	0016
Número variable Resultado de la suma (mayor el número variable) Resultado de la resta				0017 0018 0019

Por haber empleado después de la comparación de los dos números la instrucción de bifurcación BPL, dedúzcase lo que ocurre en el caso de que los dos números sean iguales.

PROYECTOS DIDACTICOS RESUELTOS CON SISTEMAS CON MICROPROCESADOR

Expuestos algunos ejemplos simples, al objeto de familiarizar al lector con el trabajo de confección de programas, a continuación se presenta una serie de proyectos de carácter puramente didáctico, en los que es necesario utilizar elementos periféricos, tales como interruptores, pulsadores, relés, diodos leds, displays, etc. Estos elementos exigen el diseño de una adaptación de los mismos con el sistema basado en el microprocesador, así como el uso del hardware auxiliar, que permita la transferencia de información entre los componentes que actúan de periféricos y la CPU.

Para la puesta en marcha de estos variados proyectos y otros, que le surgirán al lector una vez resueltos los que se proponen, se recomienda disponer de un sistema microcomputador (CPU, RAM para prueba de programas, y ROM con programa Monitor, que permita la carga y comprobación de dichos programas de trabajo) que posea entradas y salidas o permita disponerlas fácilmente. La comprobación de estos proyectos se ha llevado a cabo en módulos microcomputadores, modelo KIM-1, basados en el microprocesador R 6502 y en el entrenador de Heathkit, basado en el microprocesador Motorola 6800. (Cortesía del laboratorio de microprocesadores de ANCART.) El primero de los microcomputadores aludidos dispone de puertas de entrada y salida y el segundo puede fácilmente contar con ellas, mediante la adaptación de un PIA.

El KIM-1 está formado por un microprocesador R 6502 y dos chips R 6530, que disponen de memoria ROM en donde se guarda el programa Monitor, parte de memoria RAM, Timer y 16 pines, que permiten las entradas y salidas configuradas en dos puertas de 8 bits cada una. La figura 8-9 muestra el diagrama por bloques del KIM-1 y en ella se aprecian los dos chips R 6530, el 002 y el 003, referencias que designan la diferente programación de las ROM. El chip denominado U_2 destina sus puertas a la adaptación del microcomputador al teclado y display, así como a la posible conexión de un teletipo y un magnetofón. El U_3 deja disponibles sus puertas para configurar las entradas y salidas a través de los pines del Conector de Aplicación. También se aprecia en la figura 8-9 una memoria RAM de 1 K × 8, destinada a guardar los programas de trabajo, y un conector de Expansión para poder sacar al exterior las líneas de los buses y ampliar la potencia y capacidad del módulo. En el otro volumen de esta obra se describe con detalle la arquitectura del KIM-1, a donde se recomienda acudir al lector si lo precisa. (MICROPROCESADORES. Fundamentos, Diseño y Aplicaciones.)

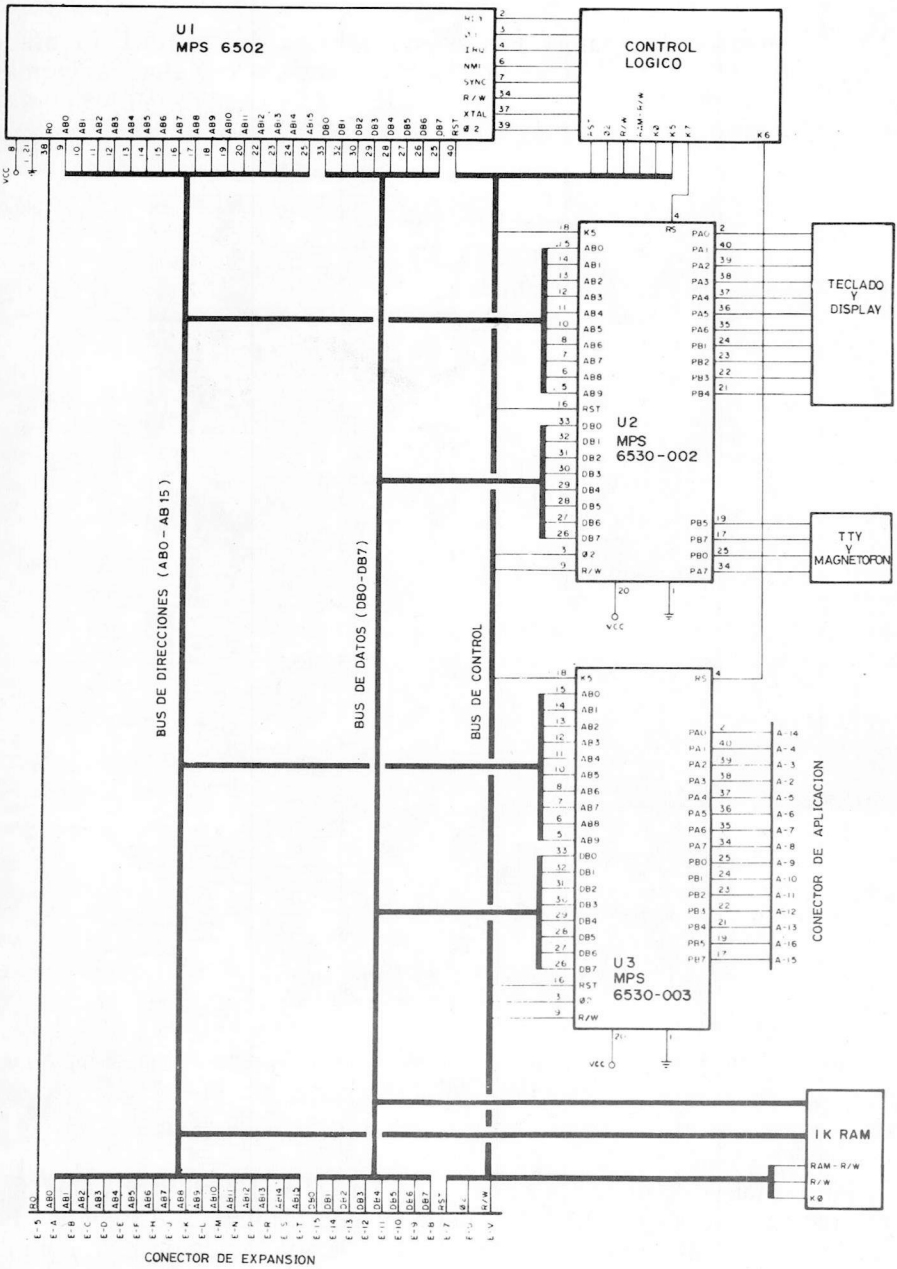


Fig. 8-9.—Diagrama por bloques del microcomputador KIM-1.

En la figura 8-10 aparece la fotografía del módulo KIM-1, montado con el conector de Aplicación por el que se introduce la alimentación de +5 V y la adaptación a un magnetofón destinado a almacenar los programas del usuario.

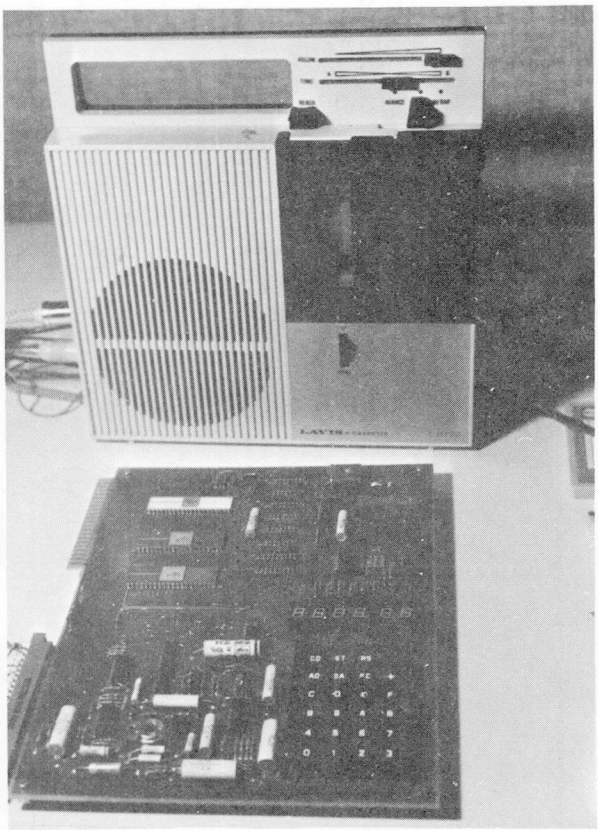


Fig. 8-10.–Módulo KIM-1 con el Conector de Aplicación que lo adapta a un magnetofón, que actúa como memoria exterior.

Básicamente el microcomputador didáctico de Heathkit dispone de la misma estructura que el KIM-1, pero sin puertas para la entrada y salida de datos, las cuales se procurarán mediante la adición de un PIA, como se representa esquemáticamente en la figura 8-11.

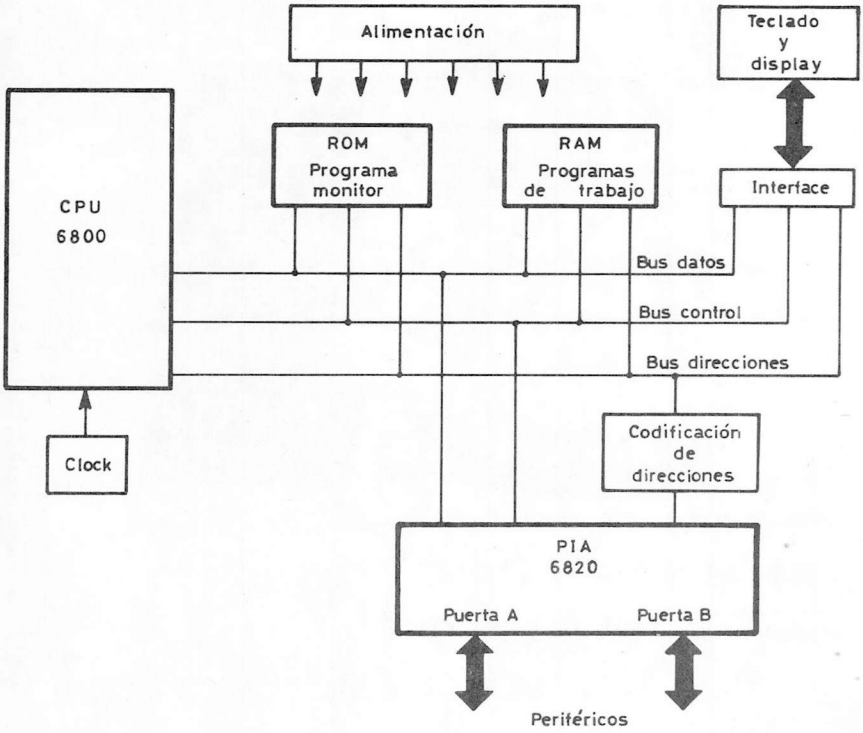


Fig. 8-11.—Diagrama esquemático del microcomputador de Heathkit y su adaptación con un PIA exterior, que permite las entradas y salidas.

Para dar una idea aproximada al montaje real de elementos auxiliares sobre el microcomputador Heathkit se ofrece en la figura 8-12 una fotografía del mismo, sobre el que se ha conectado un chip PIA, que, por medio de unos cables de interconexión, se adapta a una serie de pulsadores exteriores dispuestos sobre una caja externa que operan como elementos de entrada, y a unos leds que dispone el mismo equipo, que actúan como elementos visualizadores de salida.

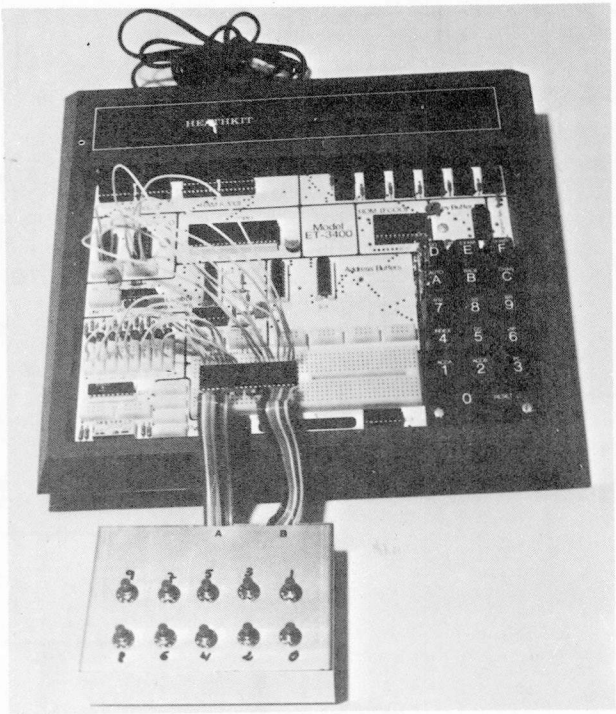


Fig. 8-12.–Foto del Heathkit y una posible adaptación, que permite el PIA.
(Cortesía de ANCART.)

1.º PROYECTO: SECUENCIA DE ENCENDIDO DE DIODOS LEDS

Planteamiento

Se trata de gobernar un conjunto de 8 leds, colocados en hilera, de forma que se vayan iluminando uno detrás de otro, produciendo una sensación de movimiento de la luz que producen.

El sentido de iluminación será de izquierda a derecha, desde el primero hasta el último, para que cuando éste se ilumine a continuación lo haga nuevamente el primero y de esta forma se repita el ciclo ininterrumpidamente.

El tiempo de iluminación de cada led vendrá determinado por una temporización (delay) establecida.

Solución con el microprocesador Motorola 6800

En principio, habrá que iluminar el primer led situado a la izquierda, mientras los restantes permanecen apagados, como aparece en la figura 8-13.

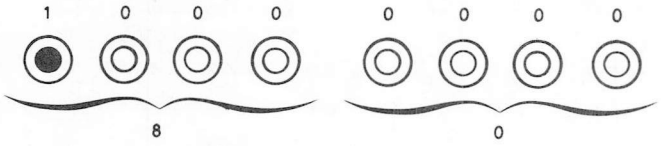


Fig. 8-13.–El led de la izquierda será el primero en iluminarse.

La activación del led de la izquierda representa que una salida de 8 bits vendrá codificada en hexadecimal por el valor 80.

Finalizado el tiempo que el 1.° led está encendido, a continuación, se encenderá el 2.° situado a su derecha, lo cual, como se representa en la figura 8-14, quedará codificado en hexadecimal por el valor 40.

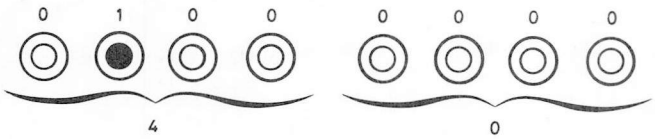


Fig. 8-14.–Encendido del 2° led y su codificación en hexadecimal.

Después del delay que mantiene encendido un cierto tiempo al 2.° led, se encenderá el 3.°, tal como aparece en la figura 8-15, con su correspondiente codificación.

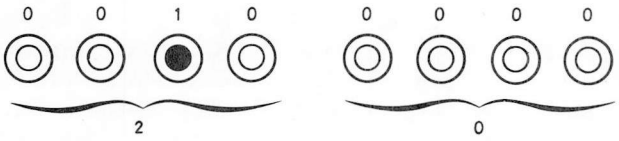


Fig. 8-15.–Encendido y codificación del 3° led.

La secuencia completa del encendido sucesivo de todos los leds se presenta en la figura 8-16-17.

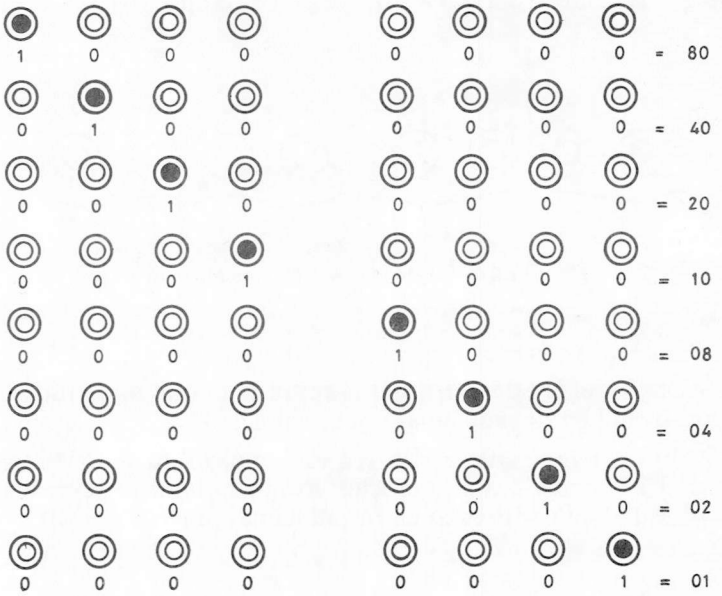


Fig. 8-16-17.– Secuencia del encendido de todos los leds.

La tabla de valores, que permite el encendido correspondiente a la secuencia de la figura 8-16, es la siguiente: 80-40-20-10-08-04-02 y 01. Esta tabla debe repetirse continuamente y de forma que, cuando finalice con el último valor, siga después con el primero. Con objeto de indicar a la CPU el final de la tabla y que ésta proceda a la iniciación de la misma, se añade otro valor cualquiera, que no corresponda con los usados, como puede serlo el 00, que actuará como valor informativo. Por este motivo la tabla quedará completada de la siguiente forma: 80-40-20-10-08-04-02-01 y 00 (valor informativo).

El organigrama que resuelva este proyecto habrá de coger el primer valor de la tabla y sacarlo por las salidas, que alimentan a los 8 leds, y permanecer en este estado mientras dure el delay conveniente: después, se toma el 2.° valor de la tabla y se procede de la misma forma, así sucesivamente, hasta que al llegar al último valor (00), se vuelva a inicializar todo

el proceso. Dicho ordinograma se presenta en la figura 8-18 y habrá de tenerse en cuenta el empleo de un PIA con el sistema microprocesador 6800, que permita la salida de los datos de la tabla por 8 líneas que activen los leds del proyecto.

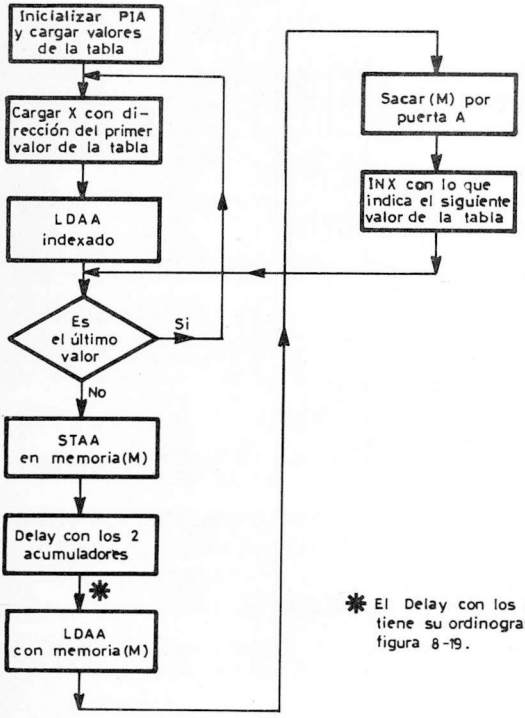
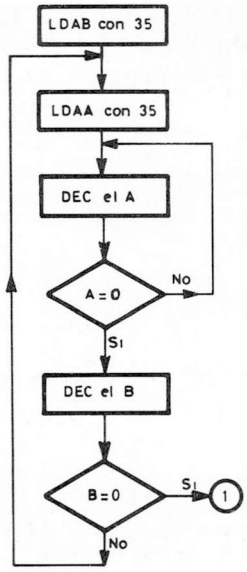


Fig. 8-18.–Ordinograma general del proyecto de encendido de leds.

El enfoque del ordinograma está preparado para acostumbrar al lector al manejo de tablas, usando el direccionamiento indexado. Existen otras formas de resolver el problema, por ejemplo, usando las instrucciones que permiten la rotación a la derecha, que, realizadas con un delay posterior, conseguirían desplazar al bit 1, que en un principio estaría situado en la posición de más peso, a través de todas las posiciones.

Previamente al delay, obsérvese que se ha almacenado el valor de la tabla, que se está trabajando en ese momento en una posición de memoria (M), al objeto de dejar libre al Acumulador A y emplearle junto con el B en la formación del delay, cuyo ordinograma se ofrece en la figura 8-19.



Nota: El valor 35 es variable y determina la duración del delay.

Fig. 8-19.–Ordinograma del delay o temporización para la permanencia del encendido de led, a base de decrementar los Acumuladores A y B.

Finalmente, para la confección del programa se destinan los 8 bits de la puerta B del PIA para actuar como salidas excitadoras de los leds, por lo que habrá que direccionar dicha puerta como salida, cargando FF (1111 1111) en la posición 8002 y 04 en la 8003.

A continuación, se presenta el programa que resuelve el proyecto de acuerdo con el ordinograma general de la figura 8-18.

Dirección	OP	Nemónico	Comentarios
0000	86 FF	LDAA ≠FF	Inicialización del PIA, puerta B; salida.
0002	B7 8002	STAA 8002	
0005	86 04	LDAA ≠04	
0007	B7 8003	STAA 8003	
000A	CE 002A	LDX 002A	Revisión de valores en la tabla.
000D	A6 00	LDAA index.	
000F	27 F9	BEQ –7	Detección de valores de la tabla.
0011	97 29	STAA en (M)	

(Continúa)

(Continuación)

Dirección	OP	Nemónico	Comentarios
0013	C6 35	LDAB ≠35	Valores para el delay que pueden variar.
0015	85 35	LDAA ≠35	
0017	27 03	BEQ +3	
0019	4A	DECA	**DELAY**
001A	20 FB	BRA −5	
001C	5A	DECB	
001D	27 02	BEQ +2	
001F	20 F4	BEQ −12	
0021	96 29	LDAA con (M)	Visualización de los valores de la tabla.
0023	B7 8002	STAA 8002	
0026	08	INX	Bucle cerrado.
0027	20 E4	BRA −31	
0029	00	Posición (M)	Posición auxiliar.
002A	80		TABLA
002B	40		
002C	20		
002D	10		
002E	08		
002F	04		
0030	02		
0031	01		
0032	00		

En la figura 8-20 se ha representado el esquema de conexionado que, a través del microcomputador, se ha montado para direccionar el PIA (Puerta B, dirección: 8002 y 8003) a base de puertas lógicas contenidas en 3 circuitos integrados. El circuito integrado SN 74 LS 241 N actúa como amplificador de las señales, que salen por la puerta B del PIA para activar a los 8 leds.

La figura 8-21 muestra la tarjeta de circuito impreso usado para implementar el hardware externo del microcomputador, en la que, además de los 8 leds, se ha añadido un microrelé conectado al 1.º led, de forma que excite y produzca el cierre de sus contactos cuando aquél se ilumine.

Solución del 1.º proyecto con el microprocesador R 6502

Se trata de resolver el proyecto de encendido secuencial de 8 leds, utilizando un sistema microprocesador basado en la familia R 6500, concretamente el sistema de desarrollo KIM-1.

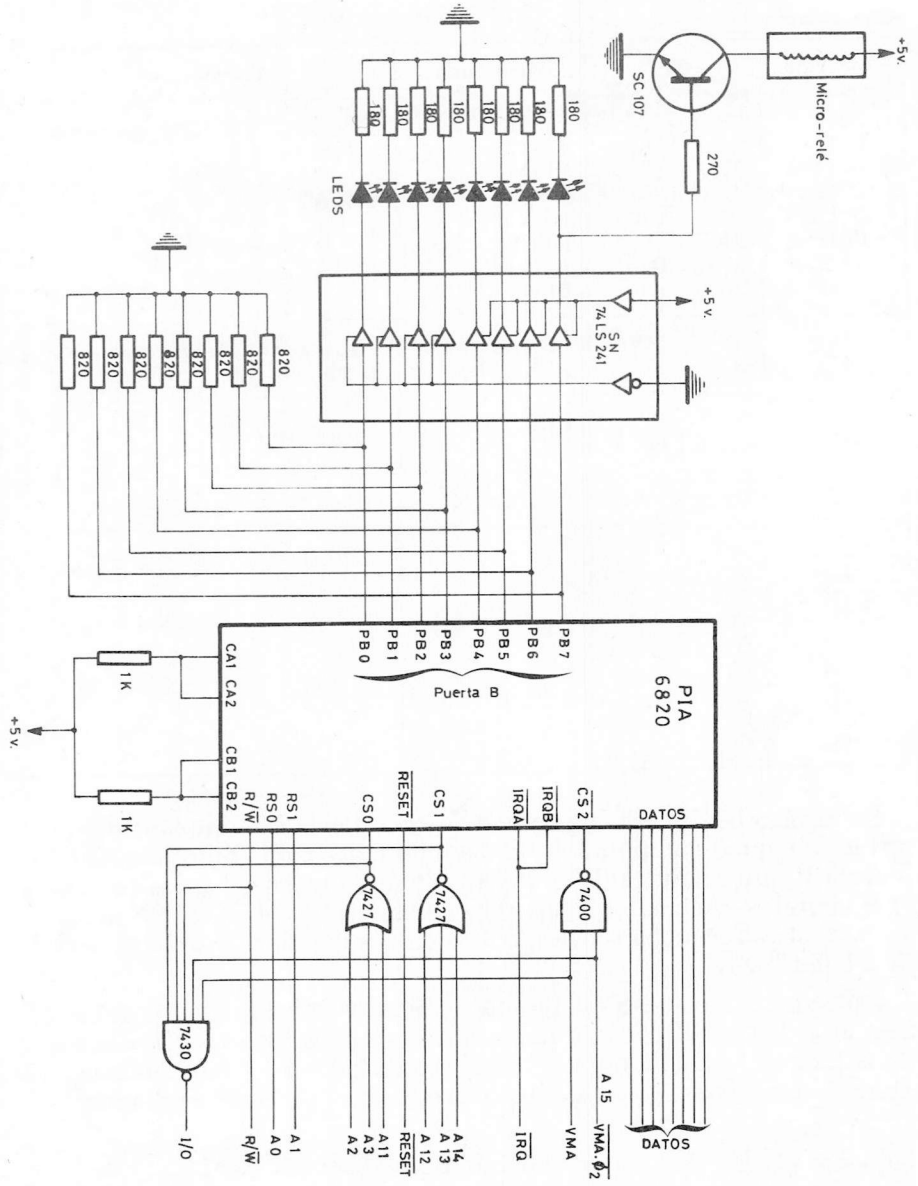


Fig. 8-20.–Esquema del conexionado del hardware externo al microcomputador. El microrelé se excita cuando se ilumina el 1.º diodo led. Las puertas que actúan sobre CS2, CS1 y CS0 ayudan a decodificar las direcciones 8002 y 8003 de la puerta B de salida de datos.

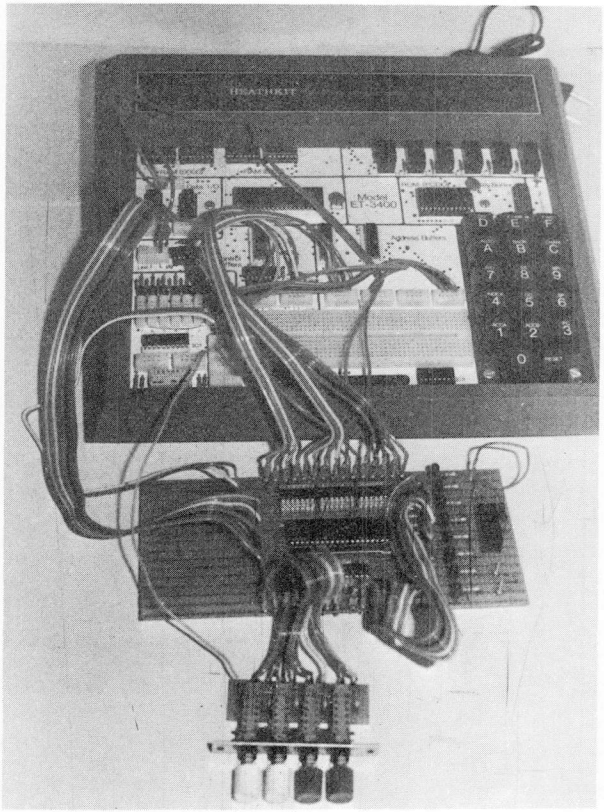


Fig. 8-21.–Fotografía del microcomputador junto con la tarjeta de circuito impreso, sobre la que se aloja el hardware auxiliar.

La única diferencia apreciable, en cuanto al ordinograma, respecto al caso anterior estriba en que, como el R 6502 sólo dispone de un Acumulador el delay, habrá que realizarlo decrementando dos posiciones de memoria. En este ejemplo se decrementa 60 veces el valor 60 hasta anularse.

En el aspecto de la inicialización y programación de la puerta de salida, que también afecta al diagrama de montaje externo, ahora se usa la puerta A del chip R 6530, que dispone el KIM-1 para aplicaciones del usuario. Responde a la dirección 1700 y se direcciona como entrada o salida mediante la 1701.

En la figura 8-22 se muestra el organigrama completo del proyecto y, a continuación, de forma resumida, el programa de instrucciones que lo resuelve en forma nemónica, usando también el código OP, preciso para introducir el programa en el KIM-1.

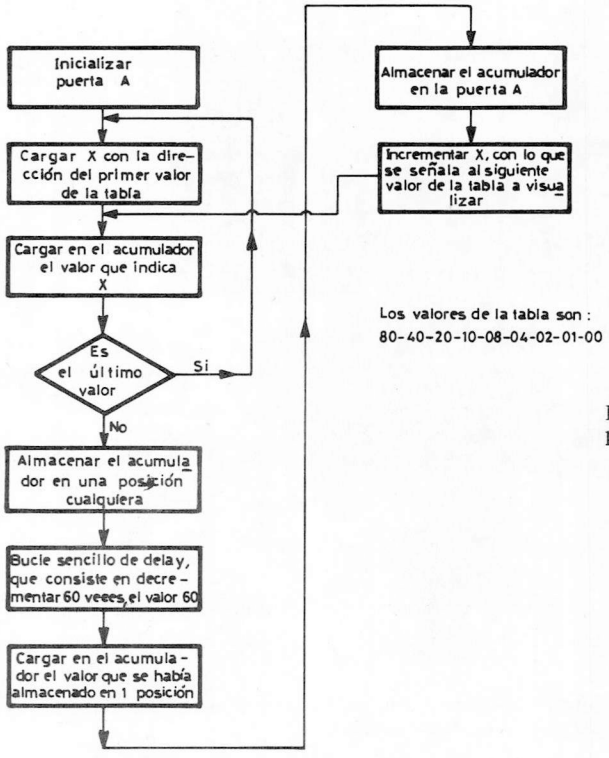


Fig. 8-22.–Ordinograma completo del proyecto para leds con R 6500.

Programa

0000	A9	FF	LDA ≠FF
0002	8D	0117	STA 1701
0005	A2	30	LDX ≠30
0007	B5	00	LDA Z pag. X
0009	F0	FA	BEQ –6
000B	85	3B	STA 3B
000D	A9	60	LDA ≠60
000F	85	39	STA 39
0011	A9	60	LDA ≠60
0013	85	3A	STA 3A
0015	C6	3A	DEC 3A
0017	A5	3A	LDA 3A

(Continúa)

(Continuación)

0019	F0	03	BEQ +3
001B	4C	1500	JMP 0015
001E	C6	39	DEC 39
0020	A5	39	LDA 39
0022	F0	03	BEQ +3
0024	4C	1100	JMP 0011
0027	A5	3B	LDA 3B
0029	8D	0017	STA 1700
002C	E8		INX
002D	4C	0700	JMP 0007
0030	80-40-20-10-08-04-02-01-00		

En la figura 8-23 se muestra un detalle de la conexión del KIM-1 con los periféricos (leds) a través de un circuito integrado, que amplifica el nivel de las señales que parten de la Puerta A del R 6530.

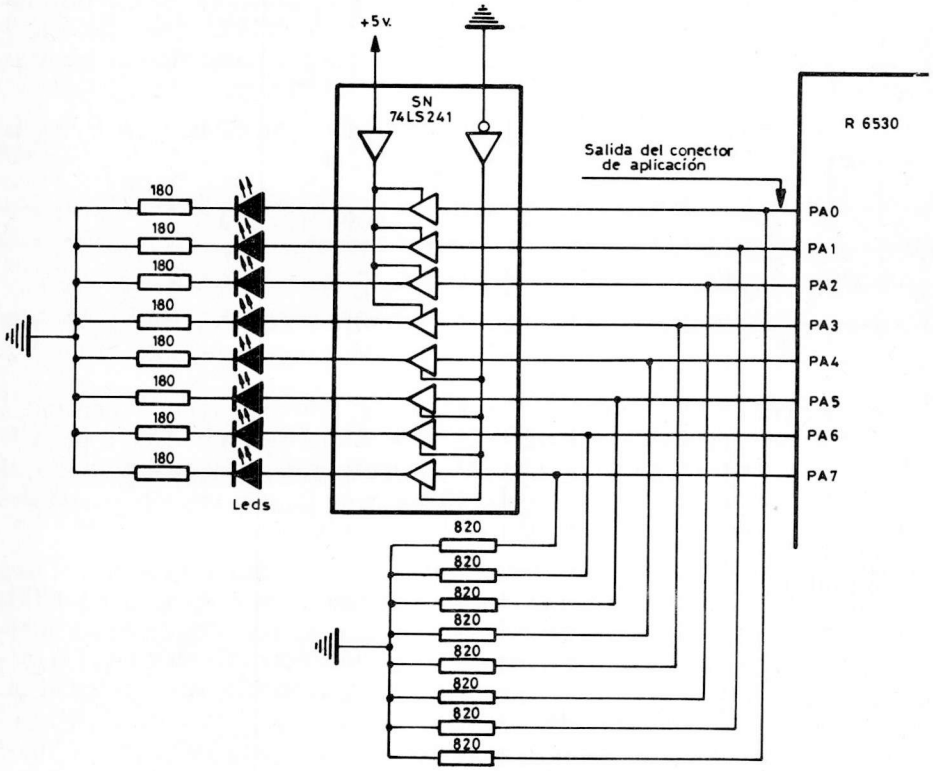


Fig. 8-23.–Detalle del circuito externo de adaptación de los leds.

2.º PROYECTO: COMPROBACION DE CIRCUITOS INTEGRADOS DIGITALES

Planteamiento:

Se trata de emplear los sistemas basados en microprocesador para la comprobación del estado y funcionamiento de las puertas, que contienen los circuitos integrados digitales.

El programa que habrá de confeccionarse deberá introducir a las entradas de las puertas diferentes niveles y procesar las salidas para conocer si realiza o no correctamente la función lógica.

Solución con el microprocesador R 6502

Con objeto de ceñir el proyecto a un determinado tipo de circuito integrado, se resuelve el mismo para su aplicación al 7402, que contiene 4 puertas NOR de 2 entradas cada una. El diseño para otro tipo de puertas sería similar.

La tabla de la verdad de una puerta NOR de 2 entradas, E_1 y E_2, es la siguiente:

E_1	E_2	Salida
0	0	1
1	0	0
0	1	0
1	1	0

De acuerdo con la tabla de la verdad, si el sistema envía a las entradas de las 4 puertas NOR dos niveles 0, las 4 salidas deberán estar a nivel 1, en caso contrario alguna de ellas estaría estropeada. Del mismo modo, si se enviasen dos niveles 1 a todas las entradas de las puertas, sus salidas deberán pasar a nivel 0 si funcionan correctamente.

Con objeto de visualizar los resultados obtenidos en las diferentes pruebas a las que se somete el circuito integrado, se dispondrá de 4 diodos luminosos (leds) conectados a las 4 salidas de las puertas. Si las puertas funcionan correctamente, los 4 leds permanecerán encendidos en todas las pruebas que se realicen. En caso de apagarse alguno, es señal de que una o más puertas son defectuosas.

En la figura 8-24 se muestra el ordinograma, que resuelve el testado y comprobación de las 4 puertas NOR del 7402.

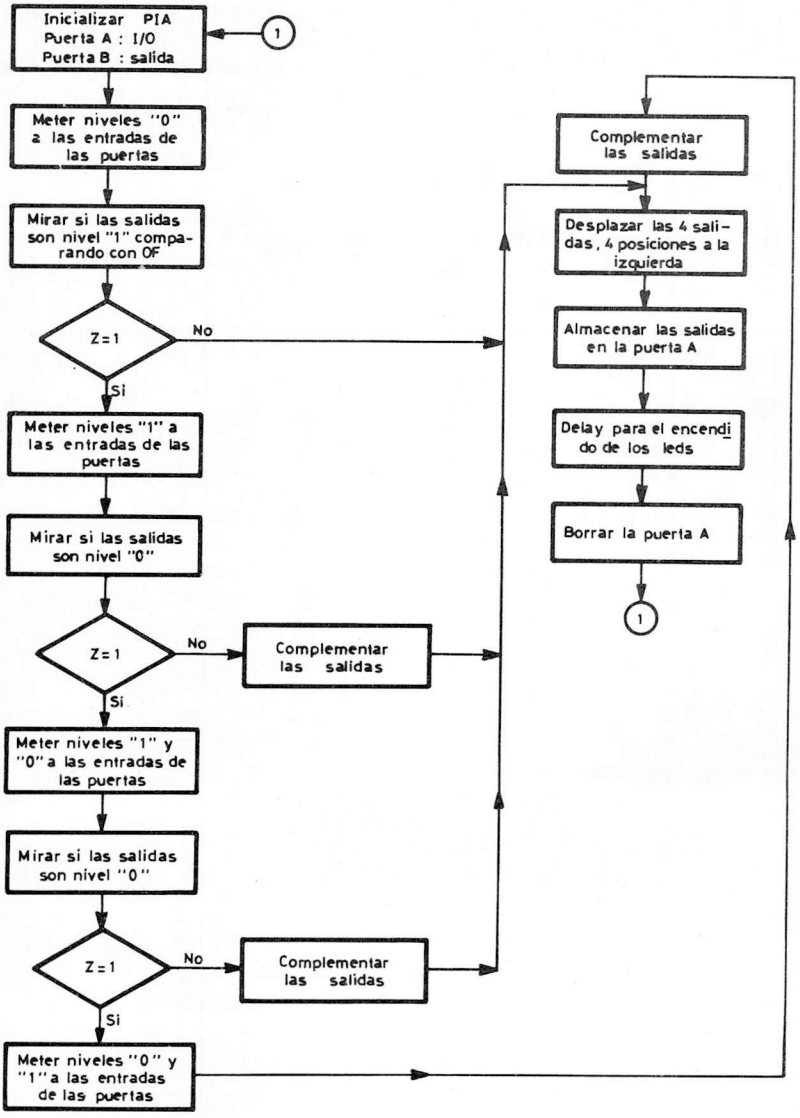


Fig. 8-24.–Ordinograma para el testado del 7402.

En el proceso de inicialización se programa a la Puerta A del R 6530 de forma que sus 4 bits de menos peso actúen como entradas a la CPU, que recogerán la información procedente de las salidas de las 4 puertas

NOR. Los 4 bits de más peso actúan como salidas del sistemas y quedan conectados a los 4 leds visualizadores del estado de las puertas, que, como ya se ha dicho, en el caso de funcionar correctamente deben de quedar iluminados. En la figura 8-25 se muestra el conexionado de la Puerta A.

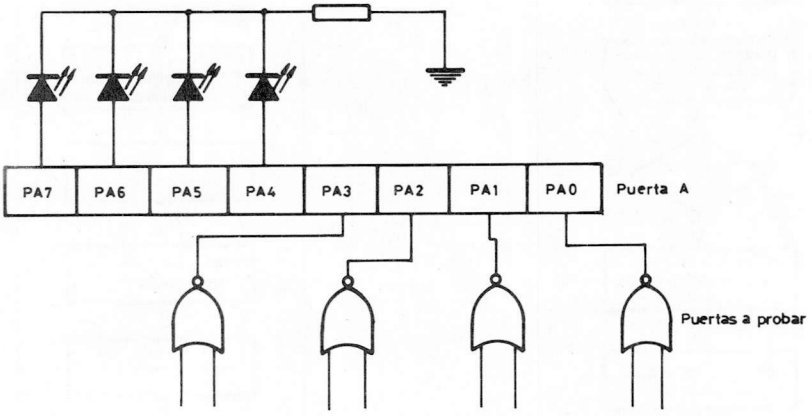


Fig. 8-25.–Conexionado de la Puerta A, actuando como elemento que permite entradas y salidas del sistema.

Por otra parte, los 2 bits de menos peso de la Puerta B se emplearán para ir introduciendo simultáneamente a las entradas de las 4 puertas las diversas combinaciones de niveles lógicos que se desean testar. (Figura 8-26.)

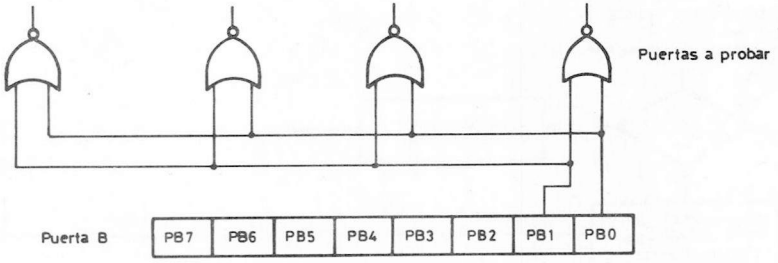


Fig. 8-26.–Conexionado de los 2 bits de menos peso de la Puerta B.

El ordinograma indica que, en un principio, por la Puerta B se aplican simultáneamente dos niveles 0 a las entradas de las 4 puertas, debiendo

producir un nivel 1 todas ellas, caso de funcionar correctamente. Para conocer si ha ocurrido así, se compara el estado de las salidas, que desde la puerta B se cargan en el Acumulador, con 0000 1111 (0F). Si el resultado de este testado o comparación no es cero (Z = 1), significa que alguna puerta NOR es defectuosa. Para conocer de cuál se trata, se procede a visualizar el estado de las salidas en los 4 leds conectados a los 4 bits de más peso de la Puerta A. A este fin se desplaza el Acumulador 4 posiciones a la izquierda y se carga en la Puerta A, con objeto de que los 4 bits de menos peso, representativos del estado de las salidas de las puertas, queden en el lugar de los 4 bits de más peso, que al pasar a la Puerta A representan las salidas de los leds, quedando apagado el correspondiente a la puerta defectuosa.

En caso de que la comparación anterior del Acumulador con el valor 0F ponga a Z = 1, se pasa a probar las puertas con otra combinación de entradas, en este caso aplicando dos niveles 1 a las entradas de las 4 puertas. El funcionamiento correcto de las NOR en esta situación producirá salidas con nivel 0, que, para realizar un tratamiento semejante al caso anterior, será necesario complementar el Acumulador que recibe la información, al objeto de que la puerta que proporcione el nivel 0 correcto cuando se visualice, ilumine el led.

En caso que las 4 puertas superen la 1.ª, 2.ª, 3.ª y 4.ª pruebas reflejadas en el ordinograma, se acaba visualizando el resultado de la 4.ª prueba en los leds, manteniendo encedidos los mismos un período de tiempo, que en el programa se ha elegido decrementando FF veces el valor FF hasta anularse.

En la figura 8-27 se muestra el esquema de conexionado de los pines del R 6530 con el circuito auxiliar exterior. Los niveles de salida de la Puerta A son insuficiente para encender los leds, por lo cual ha sido preciso amplificarlos, usando un circuito integrado buffer.

En el caso de desarrollar el proyecto sobre un microcomputador KIM-1, tal como se ha supuesto en la programación, debe tenerse presente:

- Los datos de la puerta A se manejan como si se tratase de la posición 1700 de memoria.
- Las entradas y salidas de la puerta A se determinan con el contenido de la posición 1701 (1: salida, 0: entrada).
- La puerta B responde a la posición 1702.
- El direccionado de I/O de la puerta B se realiza con el contenido de la posición de memoria 1703.

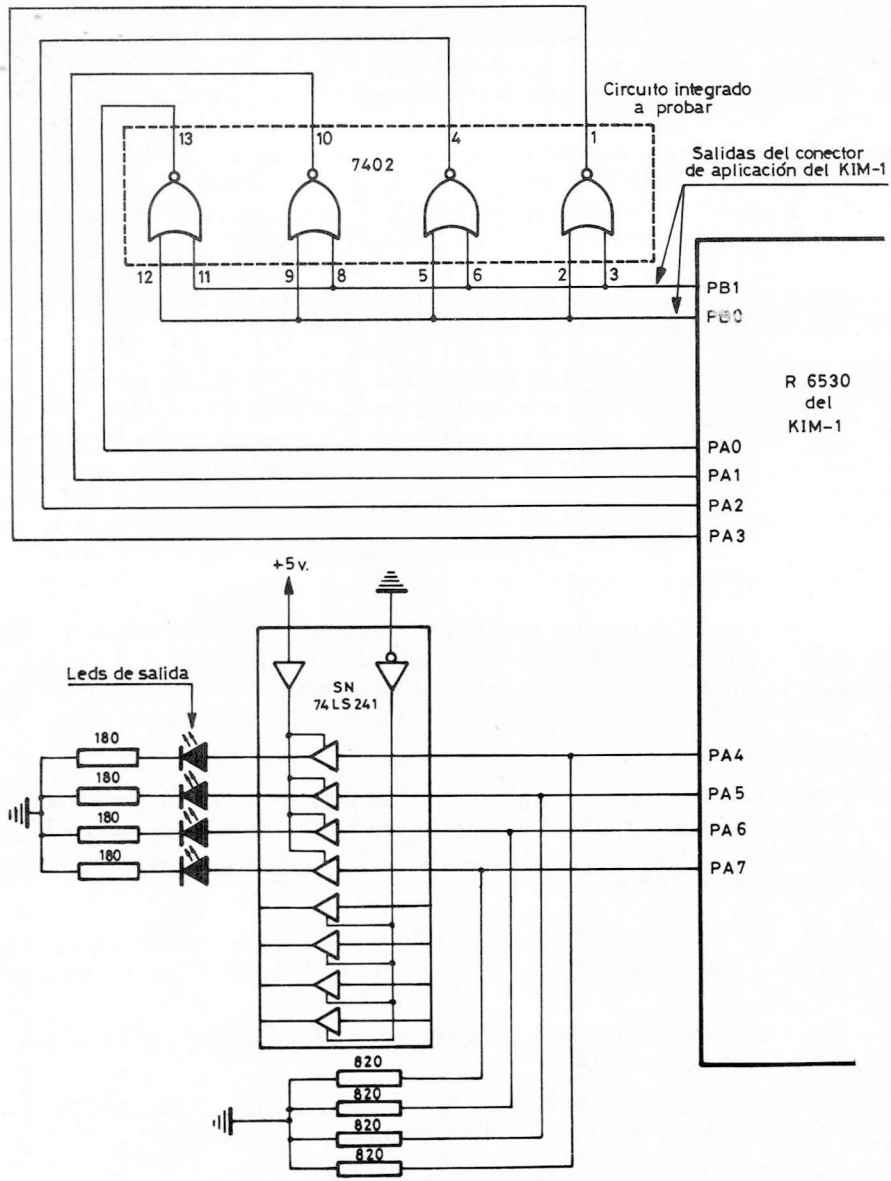


Fig. 8-27.–Conexionado del R 6530 con el circuito exterior.

El programa que responde al ordinograma de la figura 8-24 es el siguiente:

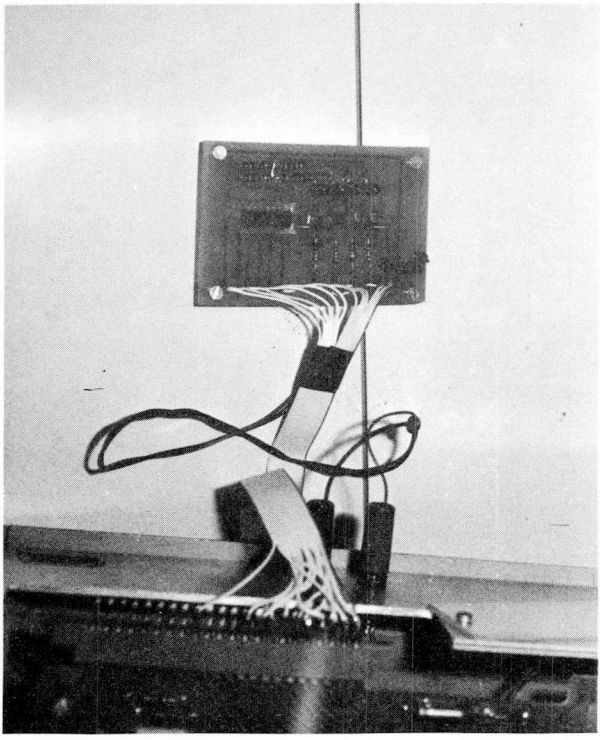


Fig. 8-27 Bis.–Fotografía de la tarjeta sobre la que se implementa el hardware externo del proyecto sobre comprobación de circuitos integrados lógicos.

PROGRAMA

0000	A9	F0	LDA ≠ F0
0002	8D	0117	STA 1701
0005	A9	03	LDA ≠ 03
0007	8D	0317	STA 1703
000A	A9	00	LDA ≠ 00
000C	8D	0217	STA 1702
000F	A9	00	LDA ≠ 00
0011	AD	0017	LDA 1700
0014	C9	0F	CMP ≠ 0F
0016	F0	03	BEQ + 3

(Continúa)

(Continuación)

0018	4C	4500	JMP 0045
001B	A9	03	LDA ≠ 03
001D	8D	0217	STA 1702
0020	A9	00	LDA ≠ 00
0022	AD	0017	LDA 1700
0025	F0	03	BEQ + 3
0027	4C	4300	JMP 0043
002A	A9	02	LDA ≠ 02
002C	8D	0217	STA 1702
002F	A9	00	LDA ≠ 00
0031	AD	0017	LDA 1700
0034	F0	03	BEQ + 3
0036	4C	4300	JMP 0043
0039	A9	01	LDA ≠ 01
003B	8D	0217	STA 1702
003E	A9	00	LDA ≠ 00
0040	AD	0017	LDA 1700
0043	49	0F	EOR ≠ 0F
0045	0A		ASL
0046	0A		ASL
0047	0A		ASL
0048	0A		ASL
0049	8D	0017	STA 1700
004C	A9	FF	LDA ≠ FF
004E	85	6E	STA 6E
0050	A9	FF	LDA ≠ FF
0052	85	6F	STA 6F
0054	C6	6F	DEC 6F
0056	A5	6F	LDA 6F
0058	F0	03	BEQ + 3
005A	4C	5400	JMP 0054
005D	C6	6E	DEC 6E
005F	A5	6E	LDA 6E
0061	F0	03	BEQ + 3
0063	4C	5000	JMP 0050
0066	A9	00	LDA 00
0068	8D	0017	STA 1700
006B	4C	0000	JMP 0000

Solución con el microprocesador Motorola 6800

A continuación se resuelve el 2.º proyecto sobre comprobación de circuitos integrados digitales, pero ahora, basándose en el microprocesador Motorola 6800 y en vez de probar el tipo 7402, se prueba el 7400, que consta de 4 puertas NAND de 2 entradas cada una.

La figura 8-28 muestra el conexionado con los periféricos.

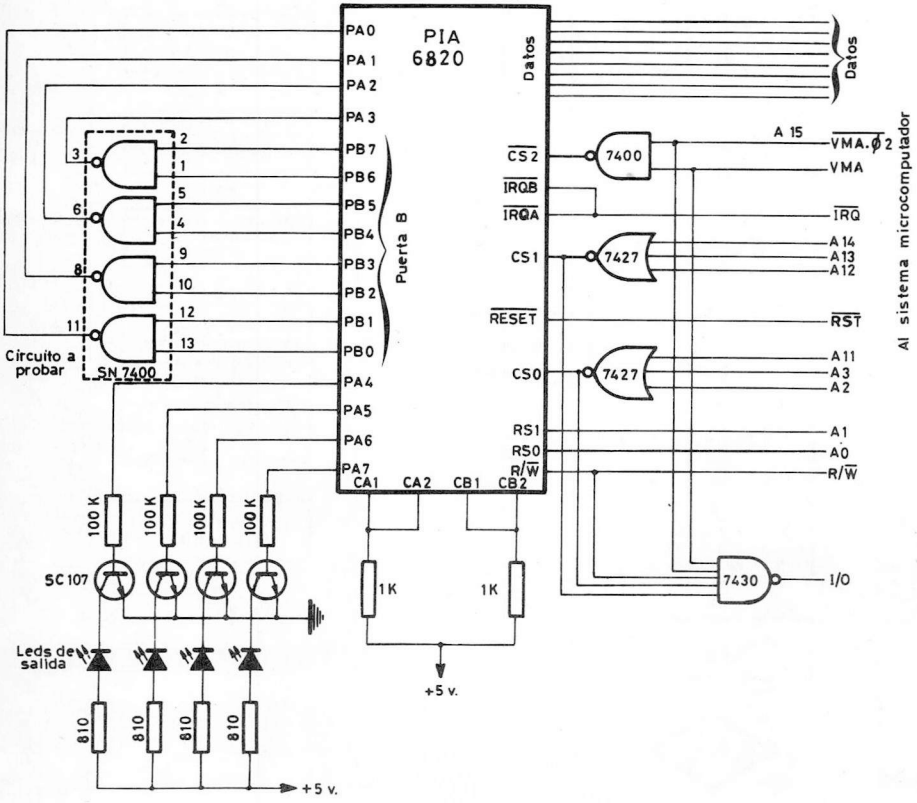


Fig 8-28.–Conexión del circuito exterior al sistema básico.

Inicializado el PIA para que los pines de sus puertas actúen como se ha indicado en la figura 8-28, el ordinograma que resuelve el proyecto es muy similar al de la figura 8-24, tal como se muestra en la figura 8-29.

Sacar por la puerta B FF, supone aplicar un nivel 1 a todos sus bits, los cuales se aplican a las entradas de las puertas NAND. Luego se comienza poniendo a 0 el Acumulador, para cargar en sus 4 bits de menos peso el estado de las salidas de las puertas (los 4 bits de más peso quedarán a nivel 0). Si las puertas NAND funcionan correctamente al recibir por sus dos entradas nivel 1, su salida será nivel 0 y el Acumulador en este caso quedará a 0. Por este motivo se testa a continuación si ha quedado a 0, comprobando el estado del flag Z (¿Z = 1?). Si este testado es afirmativo,

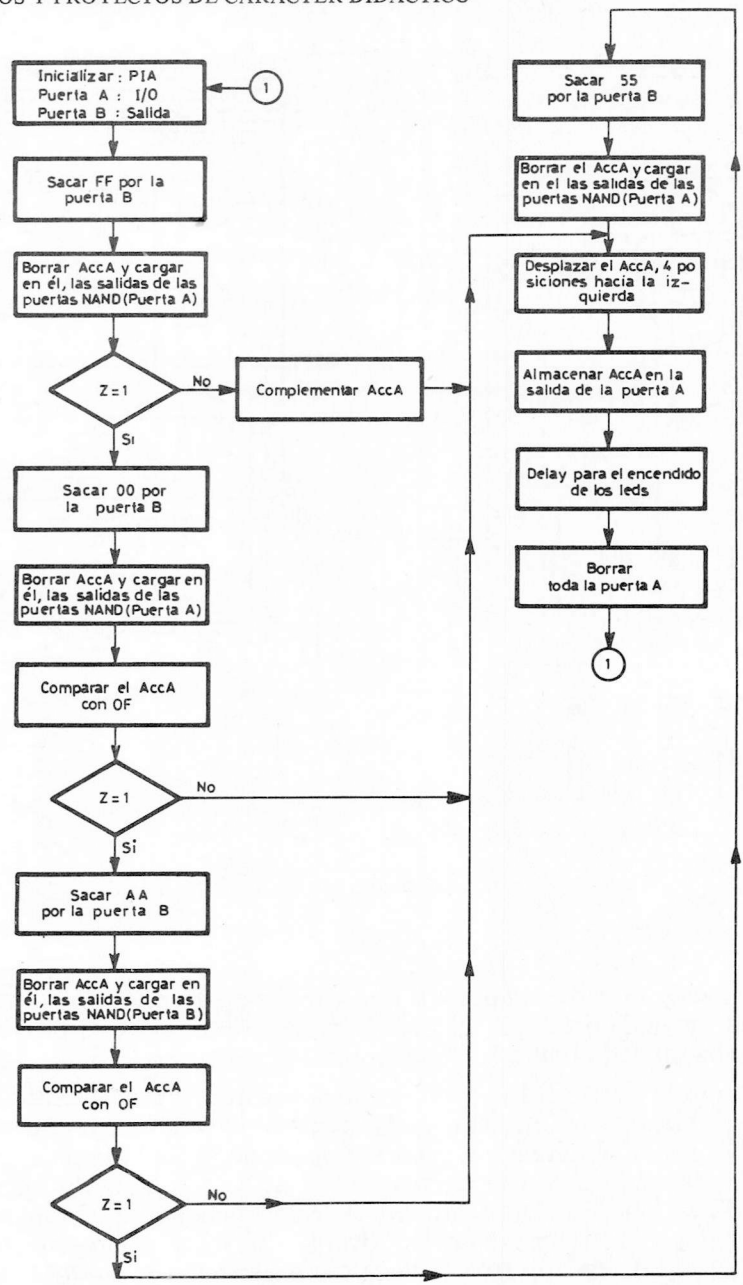


Fig. 8- 29.–Ordinograma de comprobación del 7400.

significa que las 4 puertas responden correctamente a esta 1.ª prueba, pero si, por el contrario, resulta negativo, es señal que una o varias de las puertas son defectuosas, para lo cual, después de rotar 4 posiciones a la izquierda, se visualizan las salidas (previa complementación, puesto que el nivel correcto es 0 y el encendido del led requiere nivel 1). El ordinograma provee la secuencia de pruebas para las 4 combinaciones posibles en sus dos entradas:

E_1	E_2	Salida
1	1	0
0	0	1
1	0	1
0	1	1

Como se desprende de la tabla de la verdad anterior, las salidas de las puertas NAND deben ser 1 en todos los casos menos en el 1.º, por lo que ya no se vuelve a precisar la complementación de las mismas.

Obsérvese en la figura 8-29 bis un detalle del conexionado de la puerta A, que permite comprender el efecto de las 4 rotaciones.

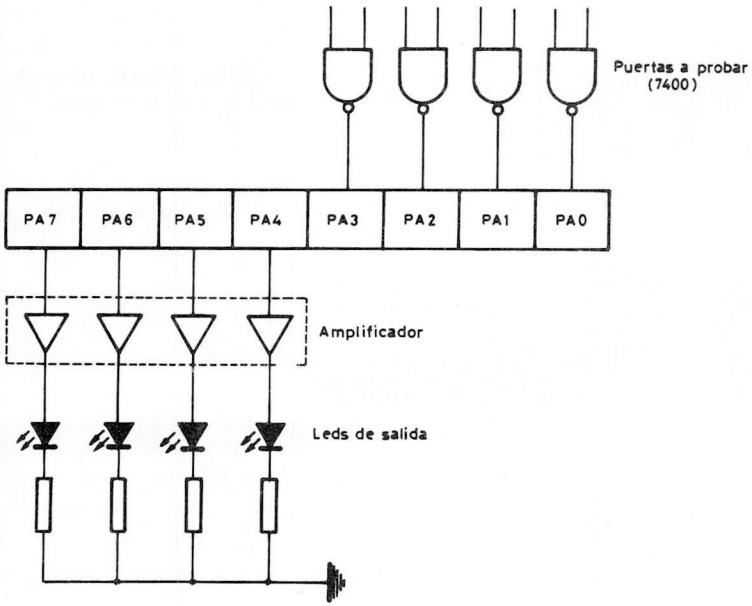


Fig. 8-29 Bis.–Conexionado de la puerta A, con 4 entradas y 4 salidas.

A continuación se presenta el programa que responde al proyecto.

0000	86F0	LDA ≠ F0	
0002	B78000	STA 8000	
0005	8604	LDA ≠ 04	
0007	B78001	STA 8001	Inicializar el PIA.
000A	86FF	LDA ≠ FF	
000C	B78002	STA 8002	
000F	8604	LDA ≠ 04	
0011	B78003	STA 8003	
0014	86FF	LDA ≠ FF	
0016	B78002	STA 8002	
0019	4F	CLR	Poner entradas de NAND a "1", viendo que su salida está a nivel cero.
001A	B68000	LDA 8000	
001D	2625	BNE +37	
001F	8600	LDA ≠ 00	
0021	B78002	STA 8002	
0024	4F	CLR	Poner las entradas de las NAND a "0",
0025	B68000	LDA 8000	viendo que sus salidas están a nivel uno.
0028	810F	CMP ≠ 0F	
002A	2619	BNE +25	
002C	86AA	LDA ≠ AA	
002E	B78002	STA 8002	
0031	4F	CLR	Poner las entradas de NAND con niveles "1" y "0", viendo que sus salidas quedan a nivel "1".
0032	B68000	LDA 8000	
0035	810F	CMP ≠ 0F	
0037	260C	BNE +12	
0039	8655	LDA ≠ 55	
003B	B78002	STA 8002	Poner entradas de NAND con niveles "0" y "1", viendo que sus salidas están a nivel "1".
003E	4F	CLR	
003F	B68000	LDA 8000	
0042	2001	BRA 01	
0044	43	COM	Complementar el acumulador
0045	48	ASL	
0046	48	ASL	Rotación del acumulador en cuatro posi-
0047	48	ASL	ciones hacia la izquierda.
0048	48	ASL	
0049	B78000	STA 8000	Visualizar el resultado de los cuatro leds.
004C	86FF	LDAA ≠ FF	
004E	C6FF	LDAB ≠ FF	DELAY para conseguir que parezca que
0050	4A	DECA	los leds están encendidos continuamente,
0051	26FD	BNE-3	indicando si hay o no error.
0053	5A	DECB	
0054	26FA	BNE-6	
0056	7F8000	CLR 8000	Borrar la posición 8000.
0059	20A9	BRA	Volver a empezar.

Nota: En todas las instrucciones se hace referencia al Acumulador A.

3.º PROYECTO: POSICIONADO DE INTERRUPTORES

En esta ocasión se pretende diseñar un sistema basado en microprocesador, que compruebe el estado de 6 interruptores y provoque la generación de una onda cuadrada de 800 c/s aproximadamente, en el caso de estar cerrados los 6. Mientras que si están abiertos, se deben producir dos niveles 1 simultáneamente, que se visualizarán con dos diodos leds.

Solución con microprocesador R 6502

Pensando implementar el sistema sobre el KIM-1, los pines PA2 a PA7 de la Puerta A del R 6530 se destinan a recibir la información sobre el estado de los 6 interruptores. Los pines PA0 y PA1 de dicha puerta servirán para sacar la onda cuadrada (el primero de ellos), cuando todos los interruptores estén cerrados y los 2 niveles 1, (ambos pines), cuando los interruptores estén abiertos.

La figura 8-30 muestra el conexionado de los elementos periféricos a la puerta del R 6530, que posee el microcomputador KIM-1 para uso desde el exterior.

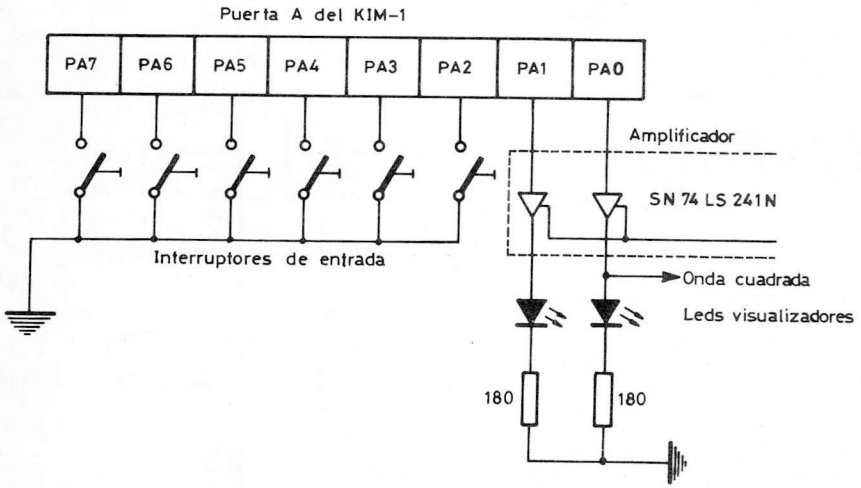


Fig. 8-30.–Conexionado de los pines de la puerta A del R 6530.

El ordinograma es muy sencillo en este caso y se limita a realizar un primer testado para conocer si los interruptores están cerrados y en caso

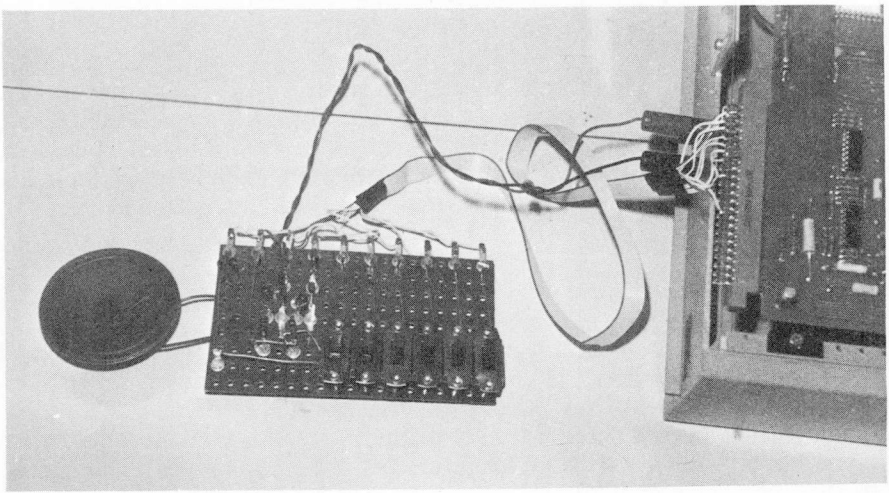


Fig. 8-30 Bis.–Detalle del montaje y conexión con el KIM, del 3º proyecto.

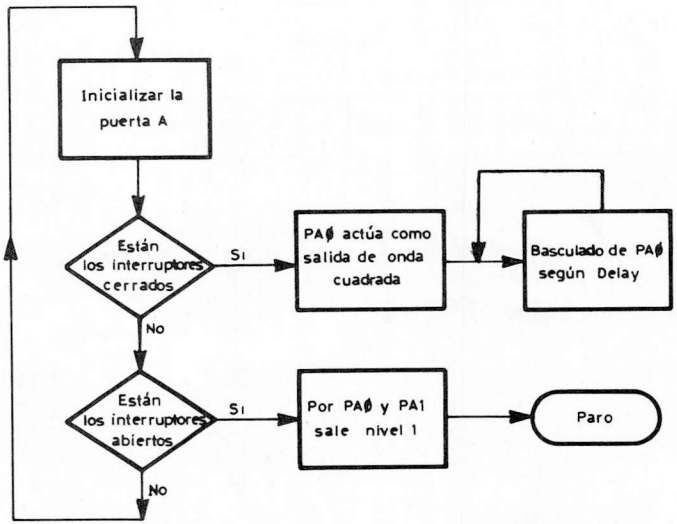


Fig. 8-31.–Duración de un ciclo cuya frecuencia aproximada es 800 c/s.

afirmativo, generar la onda cuadrada pedida. Un segundo testado para saber si están abiertos y proporcionar dos niveles 1. En la figura 8-31 se muestra el ordinograma, al que se ha hecho referencia.

Quizás el único punto del ordinograma que merezca la pena analizarse con detalle, consista en la confección del delay necesario para producir una onda cuadra de unos 800 c/s aproximadamente.

Para ello, suponemos que se trabaja con el microprocesador R 6502 a una frecuencia de 1 MHz.

El período de la onda cuadrada será:

$$T = \frac{1}{800} \simeq 0{,}001 \text{ seg.} \simeq 1.000 \text{ microseg.} \simeq 1.000 \text{ ciclos máquina.}$$

Si el período dura 1.000 μs, el basculado se producirá en la mitad de dicho tiempo, o sea, a los 500 μs, como se representa en la figura 8-32.

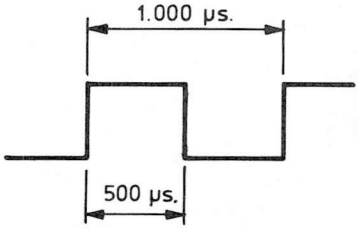


Fig. 8-32.–Duración de un ciclo cuya frecuencia aproximada es 800 c/s.

El delay se forma a base de un bucle cerrado, que va decrementando un determinado valor hasta que se anula, en cuyo momento el valor que se carga en el Acumulador se comprueba que es tal, mediante el flag Z y se sale del bucle. Las 4 instrucciones que forman dicho bucle, que más adelante pueden encontrarse en el programa y los ciclos máquina en los que se efectúa cada una, son los siguientes:

DEC ..... 5 ciclos
LDA ..... 3 ciclos
BEQ ..... 2 ciclos
JMP ..... 3 ciclos

El bucle completo representa 13 ciclos máquina y, como cada uno es de 1 μs, el tiempo que se tarda en realizar cada bucle es de 13 μs. Como el delay debe proporcionar una duración de 500 μs, el número de veces que ha de repetirse el bucle será:

$$\frac{500}{13} = 38_{10)} = 26_{16)}$$

Por último, se presenta el programa correspondiente a este proyecto.

Lenguaje ensamblador			Lenguaje máquina			
Etiqueta	Nemónico	Operando	Byte 1	Byte 2	Byte 3	Dirección
	ORG					
Start	LDA	≠ 03	A9	03		0000
	STA	Dirección A	8D	01	17	0002
	LDA	≠ 00	A9	00		0005
	LDA	Puerta A	AD	00	17	0007
	CMP	≠ 00	C9	00		000A
	BEQ	A 800 c/s	F0	0D		000C
	CMP	≠ FC	C9	FC		000E
	BEQ	A sacar "1"	F0	03		0010
	JMP	A start	4C	00	00	0012
Sacar "1"	LDA	≠ 03	A9	03		0015
	STA	Puerta A	8D	00	17	0017
	BRK		00			001A
800 c/s	LDA	≠ 01	A9	01		001B
	STA	Dirección A	8D	01	17	001D
	INX		E8			0020
	STX	Puerta A	8E	00	17	0021
	LDA	≠ 26	A9	26		0024
	STA	Posici. Delay	85	31		0026
Delay	DEC	Posici. Delay	C6	31		0028
	LDA	Posici. Delay	A5	31		002A
	BEQ	A 800 c/s	F0	ED		002C
	JMP	A Delay	4C	28	00	002E

Solución del 3.º proyecto con microprocesador Motorola 6800

La solución, en este caso, es prácticamente idéntica a la expuesta con el R 6502, únicamente que, a la hora de confeccionar el programa, en lugar de emplear el registro Indice X para producir el basculado de PAO, se utiliza el Acumulador A.

El conexionado auxiliar será similar al descrito con el chip R 6530, sólo que, en su lugar, se dispondrá de un PIA.

El programa, usando las instrucciones y códigos del microprocesador M 6800, es el siguiente:

0000	86	03	LDAA ≠ 03
0002	B7	8000	STAA 8000
0005	86	04	LDAA ≠ 04
0007	B7	8001	STAA 8001
000A	B6	8000	LDAA 8000
000D	81	00	CMPA ≠ 00
000F	27	0C	BEQ +13
0011	81	FC	CMPA ≠ FC

(Continúa)

(Continuación)

0013	27	02	BEQ +2
0015	20	E9	BRA −23
0017	86	03	LDAA ≠ 03
0019	B7	8000	STAA 8000
001C	3E		HLT
001D	86	01	LDAA ≠ 01
001F	B7	8000	STAA 8000
0022	86	04	LDAA ≠ 04
0024	B7	8001	STAA 8001
0027	4C		INCA
0028	B7	8000	STAA 8000
002B	C6	26	LDAB ≠ 26
002D	5A		DECB
002E	27	F7	BEQ −9
0030	20	FB	BRA −5

4.º PROYECTO: ORGANO ELECTRONICO

Como continuación del proyecto anterior se expone ahora otra aplicación real, que se diseña sólo para el microprocesador R 6502, contenido en el microcomputador KIM-1 y que consiste en la generación de diferentes ondas cuadradas, de frecuencia variable y audible mediante un altavoz, que se producen según los interruptores que se cierren.

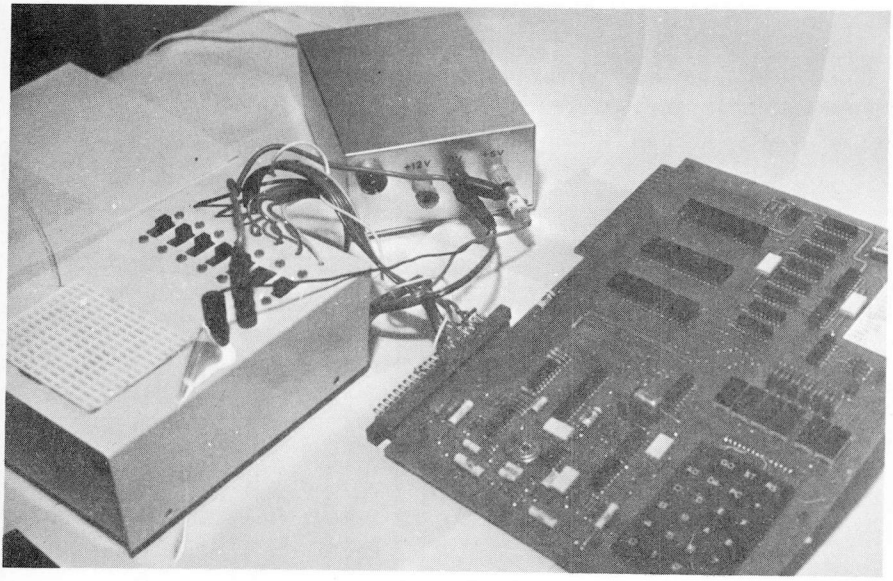


Fig. 8-33.–Foto del órgano electrónico. En la caja auxiliar se pueden apreciar los interruptores. Interiormente se encuentra el altavoz con su circuito de activación. (Cortesía de ANCART.)

En la figura 8-33 se muestra una fotografía del diseño presente, preparado para su adaptación al KIM-1.

La conexión de los interruptores de entrada y la salida al altavoz desde la puerta A del R 6530 se muestra en la figura 8-34.

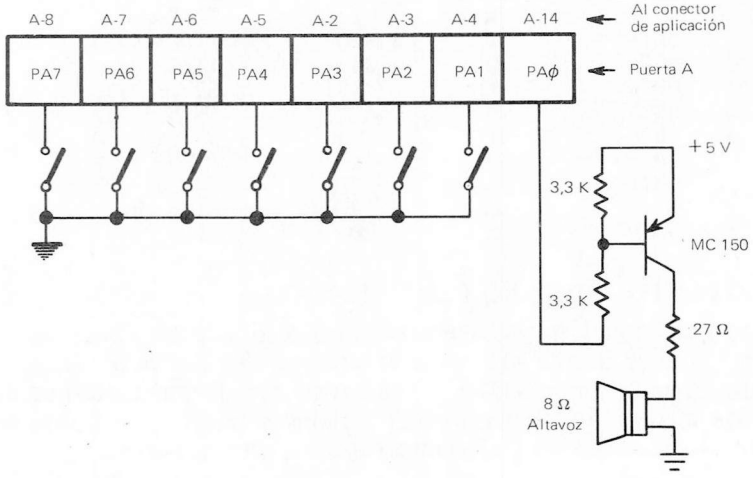


Fig. 8-34.–Conexión de los periféricos a la puerta A del R 6530.

El ordinograma, que corresponde al proyecto del órgano electrónico, se muestra, de forma esquemática, en la figura 8-35.

A continuación se presenta el programa de instrucciones, que resuelve el anterior ordinograma en lenguaje Ensamblador.

Etiqueta	OP Código	Operando	Ciclos Máquina	Comentarios
INIT	LDA	# $01	2	Define I/O 0: Entrada 1: Salida
	STA	PADD	4	PADD: P: Puerta A: Registro
START	INC	PAD	6	D: Dirección D: Datos
READ	LDA	PAD	4	Cambio de PA ϕ. No afecta a las
	EOR	# $FF	2	entradas PA - 1 a PA - 7. Lee los
	LSR	A	2	interrp. en el Acum. Complementa
	TAX		2	su valor. Desplaza a la der. 1 bit el
DELAY	DEX		2	Acum. Transfiere el contaje a X.
	BPL	DELAY	3,2	Retardo según el valor especificado en el Indice X.

(Continúa)

(Continuación)

Etiqueta	OP Código	Operando	Ciclos Máquina	Comentarios
	BMI	START	3	Va a Start
PADD	= $1701			Define la dirección absoluta del Reg. A de dirección Datos.
PAD	= $1700			Define la direcc. absoluta del Reg. A

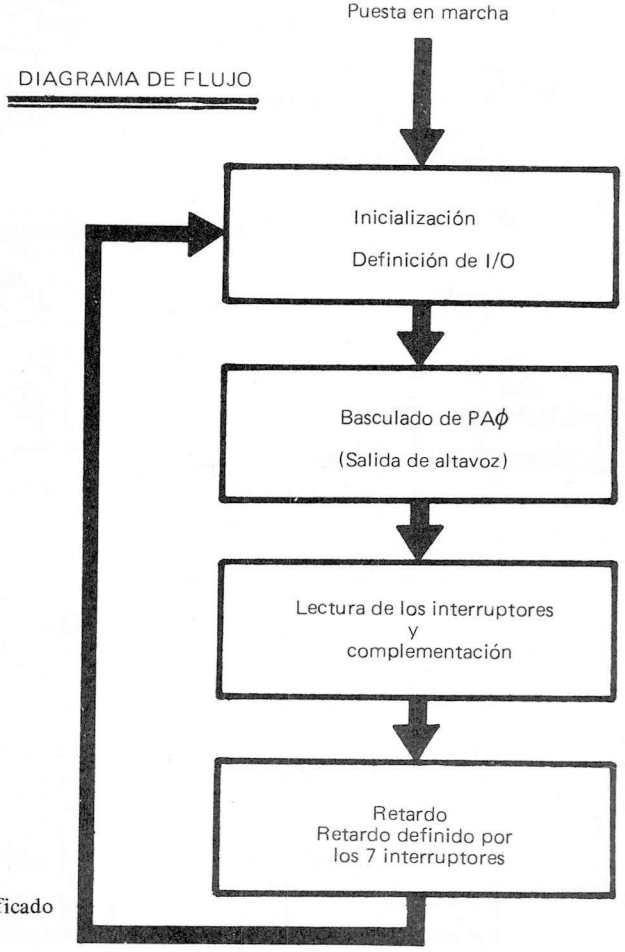
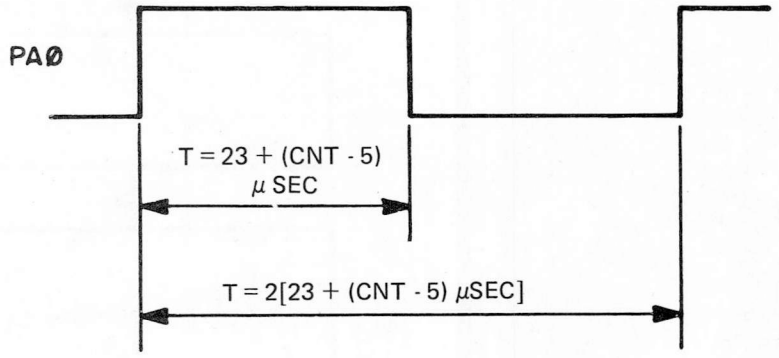


Fig. 8-35.—Ordinograma simplificado del órgano electrónico.

Cuando se utiliza el lenguaje Ensamblador, cada línea del programa se divide en varios campos, que se explican a continuación y que son semejantes a los explicados en el capítulo 6.

- *"Campo de etiqueta"*, que sirve para asignar un "nombre" a una posición específica del programa.
- *"Campo del Código OP"* (Código de Operación), en el que se define la instrucción exacta, que ha de ejecutarse con nemónicos.
- *"Campo de operando"* donde se definen los datos que precisa la instrucción, junto a ciertos símbolos que definen el modo de direccionado, así como el formato o código en que vienen expresados los datos. Los símbolos más destacados son:

  ≠ Direccionamiento inmediato.
  $ Código hexadecimal.
  C Código octal.
  % Código binario.
  ' ASCII literal.
  = Iguala una etiqueta a un valor o posición de memoria.
- *"Campo de Ciclo máquina"*, que define el número total de ciclos máquina que se requieren para ejecutar una instrucción. Normalmente no se emplea.



$$FREQ = \frac{1}{T} = \frac{10^6}{46 + 10 \cdot CNT} \text{ CPS}$$

Fig. 8-36.–Onda cuadrada y determinación de su frecuencia.

NOTA: CNT equivale al valor del registro Indice X, que es función del estado de los 7 interruptores, $0 \leqslant CNT \leqslant 127$

– *"Campo de Comentarios"*, donde el programador explica el objeto de las instrucciones que emplea.

Para determinar la frecuencia de la onda cuadrada, variable según el estado o lectura de los interruptores, se parte del total de ciclos maquina, que suceden entre cada cambio de nivel de PAO, de lo que se puede deducir una ecuación para determinar la frecuencia de la onda cuadrada. La figura 8-36 muestra la forma de la onda obtenida por PAO y las ecuaciones que sirven para determinar su frecuencia.

En la figura 8-36, CNT equivale al valor del registro Indice X, el cual es función del estado o lectura de los 7 interruptores, de forma que $0 \leqslant \leqslant CNT \leqslant 127$, puesto que el valor mayor será $111\ 1111 \equiv 127$.

Finalmente, se proporciona el programa correspondiente al órgano electrónico en Código Máquina, para poder cargarlo en la memoria RAM del KIM-1 una vez implementado el circuito externo.

Dirección	INSTRUCCIONES			CODIGO FUENTE		
	Byte 1	Byte 2	Byte 3	Etiqueta	OP Código	Operando
ϕ2ϕϕ	A9	ϕ1		INIT	LDA	# $01
ϕ2ϕ2	8D	ϕ1	17		STA	PADD
ϕ2ϕ5	EE	ϕϕ	17	START	INC	PAD
ϕ2ϕ8	AD	ϕϕ	17	READ	LDA	PAD
ϕ2ϕB	49	FF			EOR	# $FF
ϕ2ϕD	4A				LSR	A
ϕ2ϕE	AA				TAX	
ϕ2ϕF	CA			DELAY	DEX	
ϕ21ϕ	1ϕ	FD			BPL	DELAY
ϕ212	3ϕ	F1			BMI	START
ϕ214						

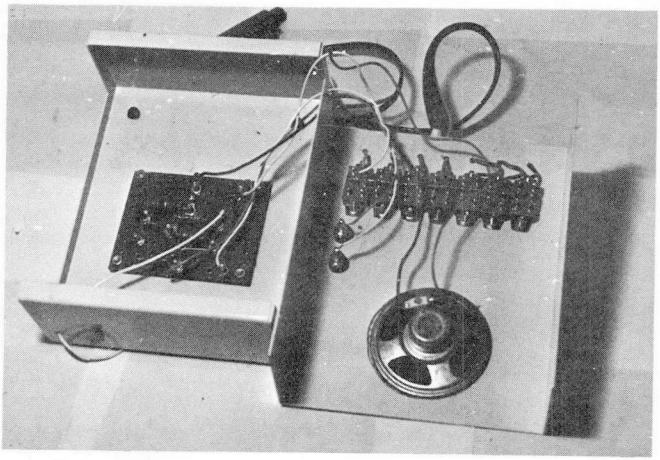


Fig. 8-37.–Foto de hardware externo y periférico (interruptores y altavoces) del órgano electrónico.

5.º PROYECTO: CONTADOR DE SUCESOS Y PROGRAMA

Se pretende diseñar un generador de impulsos, mediante un sistema con microprocesador, que, acoplado a un contador, permita conocer el número de ciclos o impulsos que se han producido en un espacio de tiempo. Con este proyecto se trata de controlar otros sucesos o programas, que puedan ocurrir en un módulo microcomputador y visualizar su número mediante un contador auxiliar de 3 dígitos.

Además de la visualización de los impulsos en el contador, existirá un registro interno en el que se guarde también dicho valor.

Solución con microprocesador R 6502

En la figura 8-38 se presenta un esquema simplificado de la conexión de la unidad contadora al KIM-1, donde se ha realizado el montaje. En la foto de la figura 8-39 se ofrece una vista del diseño presente.

El programa se basa fundamentalmente en dos delays o temporizaciones. El primero regula la frecuencia en que se producen los impulsos, mientras que el segundo regula el tiempo que transcurre entre el flanco ascendente y el descendente (véase la figura 8-40).

La parte señalada como "a" es el tiempo que transcurre entre dos impulsos sucesivos y su duración está controlada por el primer delay. La zona "b" expresa el tiempo que dura el impulso, o sea, que se produce el nivel 1, y queda controlado mediante el segundo delay.

El primer delay consiste en la repetición de 3 bucles. El primero de ellos decrementa FF veces el valor FF, hasta anularlo. Al acabar este bucle, se pasa al siguiente que realiza la misma función y, cuando este 2.º

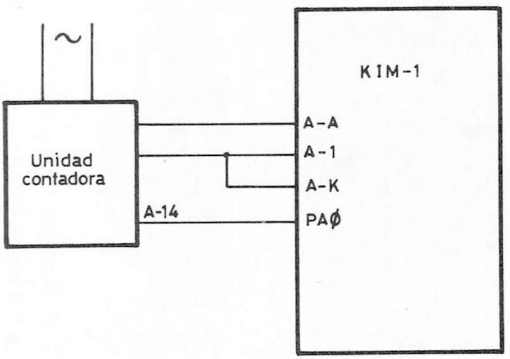


Fig. 8-38.–Conexión entre la unidad contadora y el KIM.

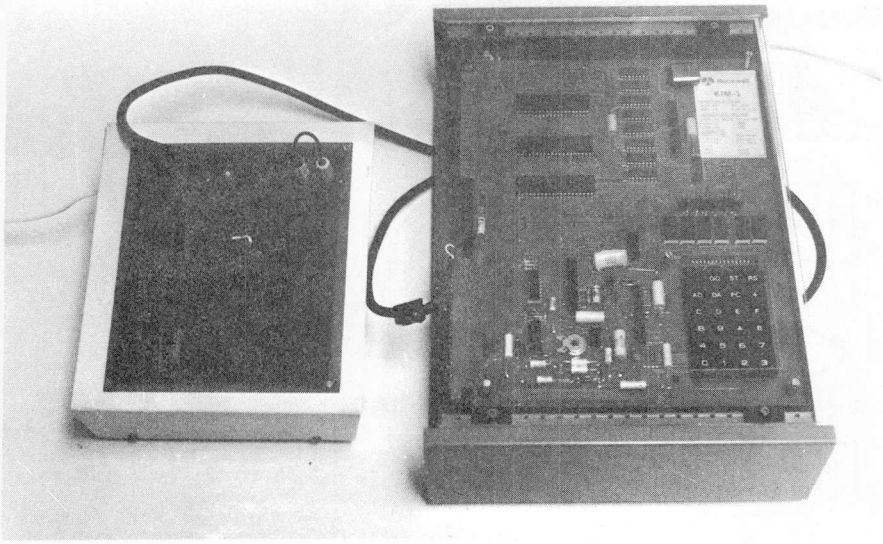


Fig. 8-39.–Fotografía del diseño del contador acoplado al KIM-1.

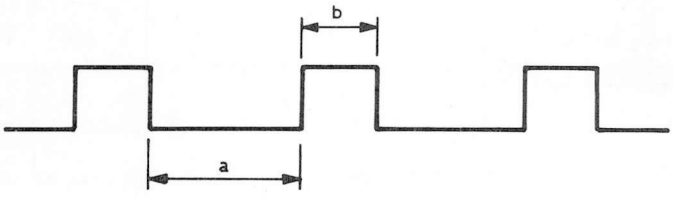


Fig. 8-40.–Regulación de los dos tiempos en los que se produce el ciclo.

bucle termina, comienza un 3.º que se encarga de hacer repetir los dos bucles anteriores tantas veces como indique el valor que se escoja previamente. Así, si dicho valor es FF, los dos primeros bucles habrán de repetirse FF veces, con lo que se alcanza la máxima temporización que es de unos 8 minutos.

Al final del 1.º delay la salida PAO debe alcanzar el nivel 1, el cual permanece el tiempo que regula el 2.º delay. Este 2.º delay consiste sencillamente en decrementar un número de veces, por ejemplo 30, el valor FF, produciéndose al término del mismo el basculado de PAO al nivel 0.

La figura 8-41 muestra la realización de un bucle sencillo.

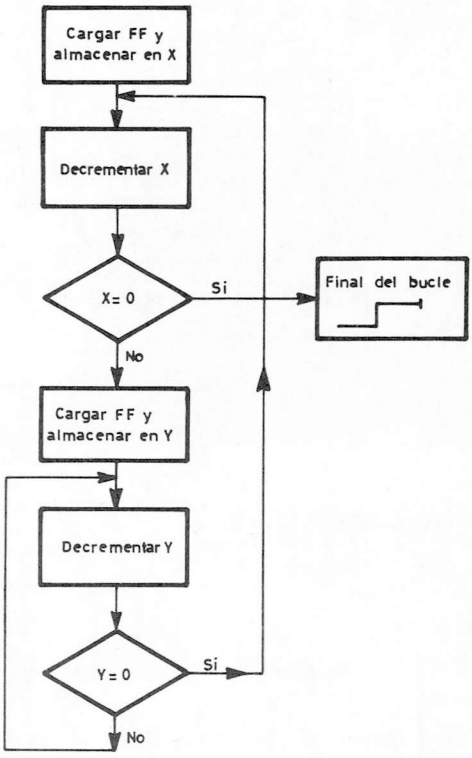


Fig. 8-41.—Organigrama de un delay para decrementar FF veces el número FF hasta que se anula. X e Y son dos posiciones de memoria.

El ordinograma del programa completo del 5.º proyecto se presenta en la figura 8-42.

El listado completo del programa que responde al ordinograma de la figura 8-42 se presenta a continuación.

```
0000    A900      LDA ≠ 00
0002    856B      STA 6B
0004    A901      LDA ≠ 01
0006    8D0117    STA 1701
0009    A9FF      LDA ≠ FF
000B    8568      STA −68
000D    A900      LDA ≠ 00
000F    8D0017    STA 1700
0012    A9FF      LDA ≠ FF
0014    8569      STA 69
```

* El contenido de la posición 000A variará el retraso con que se produzcan los impulsos.

(Continúa)

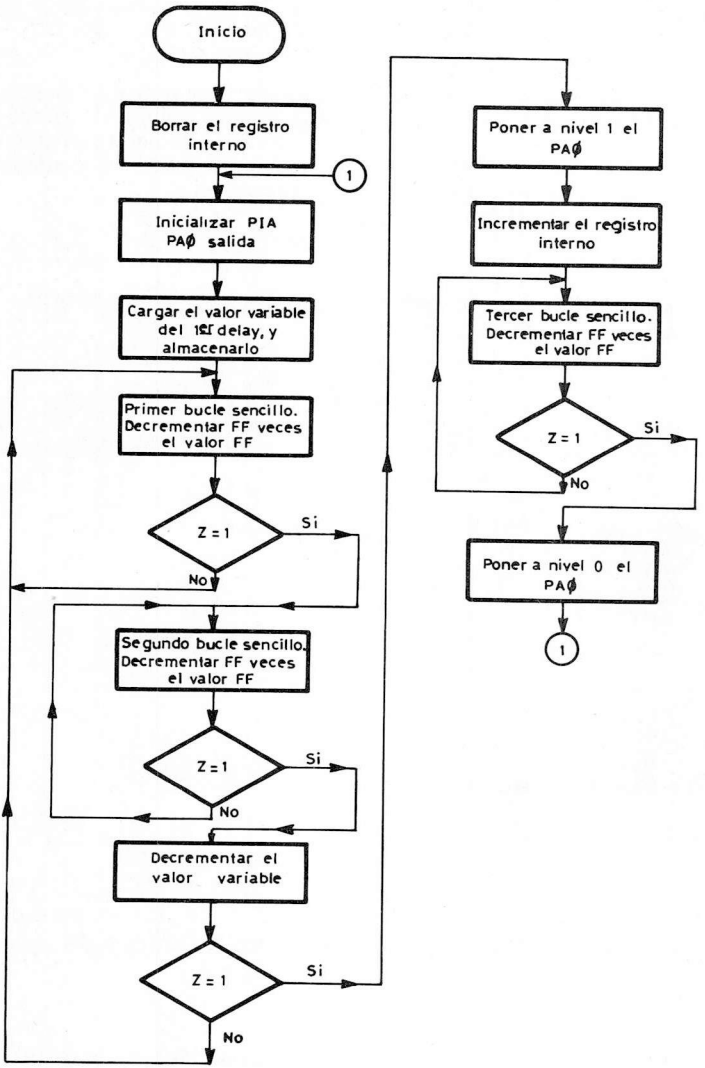


Fig. 8-42.–Ordinograma completo del generador de impulsos.

(Continuación)

0016	C669	DEC 69
0018	A569	LDA 69
001A	F00D	BEQ +13
001C	A9FF	LDA ≠ FF

* En la posición 006B tendremos el número de impulsos que se han realizado desde el principio del programa.

(Continúa)

(Continuación)

001E	856A	STA 6A
0020	C66A	DEC 6A
0022	A56A	LDA 6A
0024	F0F0	BEQ −16
0026	4C2000	JMP 0020
0029	A9FF	LDA ≠ FF
002B	8569	STA 69
002D	C669	DEC 69
002F	A569	LDA 69
0031	F00D	BEQ +13
0033	A9FF	LDA ≠ FF
0035	856A	STA 6A
0037	C66A	DEC 6A
0039	A56A	LDA 6A
003B	F0F0	BEQ −16
003D	4C3700	JMP 0037
0040	C668	DEC 68
0042	A568	LDA 68
0044	F003	BEQ +3
0046	4C1200	JMP 0012
0049	A901	LDA ≠ 01
004B	8D0017	STA 1700
004E	E66B	INC 6B
0050	A930	LDA ≠ 30
0052	8569	STA 69
0054	C669	DEC 69
0056	A569	LDA 69
0058	F0AB	BEQ −85
005A	A9FF	LDA ≠ FF
005C	856A	STA 6A
005E	C66A	DEC 6A
0060	A56A	LDA 6A
0062	F0F0	BEQ −16
0064	4C5E00	JMP 005E

* Puede disminuirse la longitud del programa realizando el testado de Z = 1 directamente en la posición decrementada, sin pasar su contenido por el Acumulador.

La figura 8-43 ofrece el esquema que se ha utilizado para el montaje del contador empleado, cuya foto aparece, en la figura 8-44.

Solución del 5.º proyecto con microprocesador Motorola 6800

La interconexión necesaria entre el sistema básico con el microprocesador y la unidad contadora a través del PIA se muestra en la figura 8-45.

La única variante en cuanto a la realización del programa con las instrucciones del 6800, es que los delays se han confeccionado empleando los dos Acumuladores A y B del microprocesador, realizándose los decre-

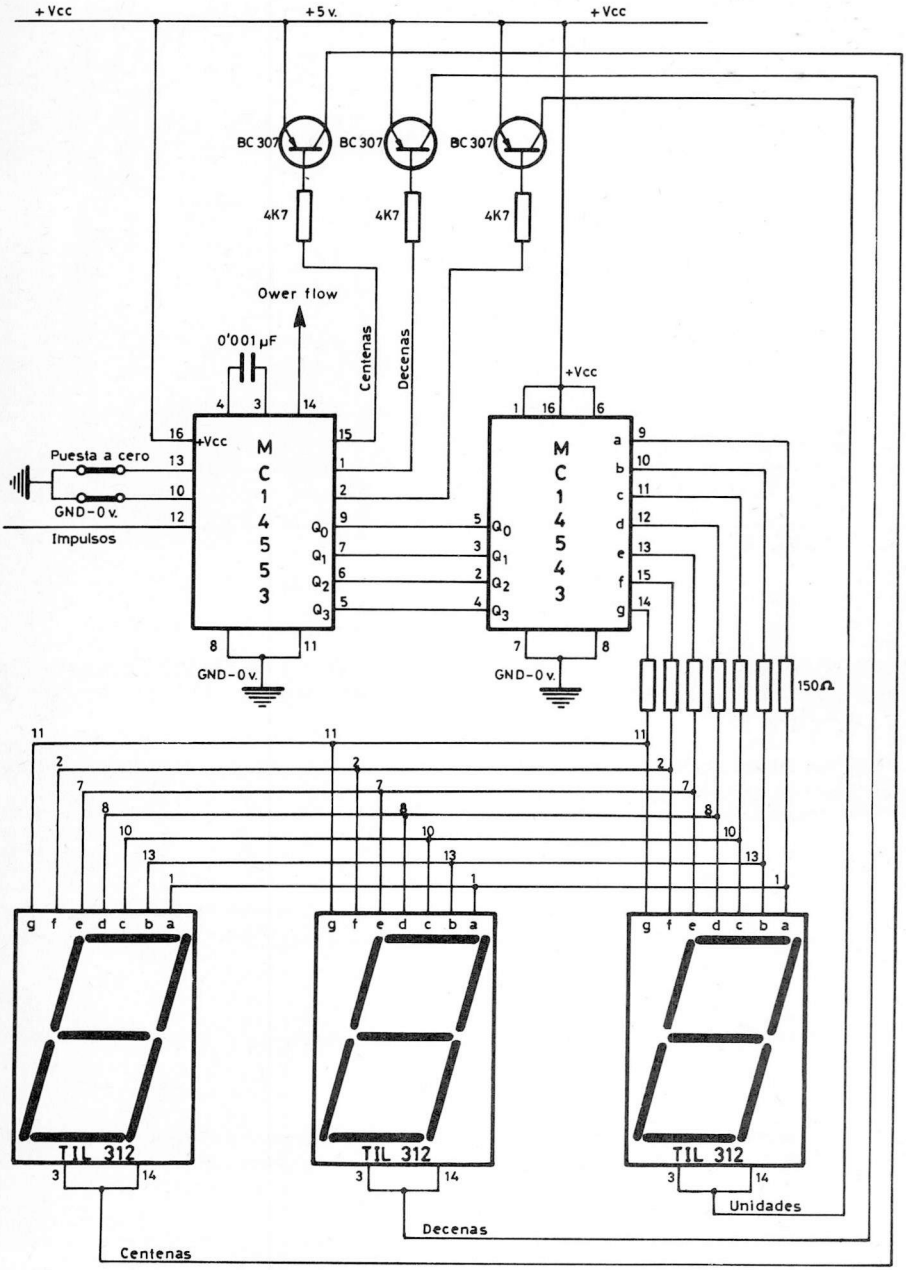


Fig. 8-43.–Esquema de la unidad contadora.

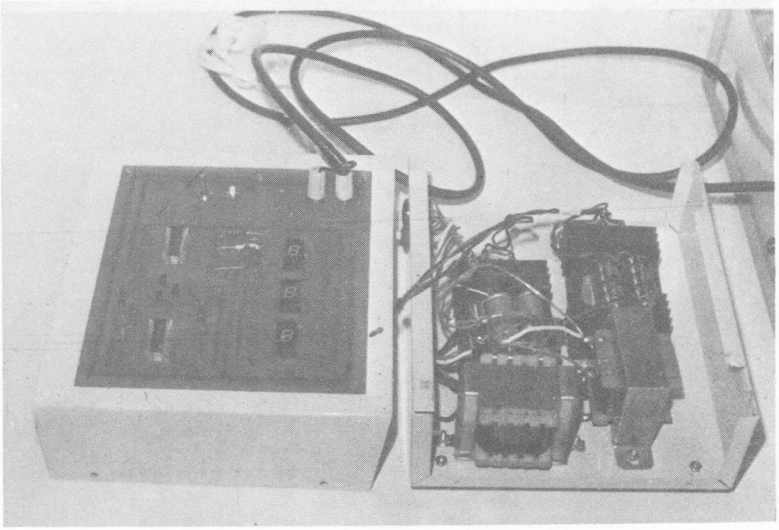


Fig. 8-44.–Fotografía de la unidad contadora.

mentos sobre ellos y evitando la utilización de posiciones de memoria para este fin. La figura 8-46 muestra el ordinograma correspondiente a uno de estos bucles sencillos.

Por último, se presenta el programa de instrucciones correspondiente a este diseño, empleando las instrucciones del 6800.

PROGRAMA

0000	86	00	LDAA ≠ 00
0002	97	4C	STAA en 4C
0004	86	01	LDAA ≠ 01
0006	B7	8000	STAA en 8000
0009	86	04	LDAA ≠ 04
000B	B7	8001	STAA en 8001
000E	86	FF	LDAA ≠ FF
0010	97	4D	STAA en 4D
0012	86	00	LDAA ≠ 00
0014	B7	8000	STAA en 8000
0017	86	FF	LDAA ≠ FF
0019	4A		DECA
001A	27	07	BEQ +7
001C	C6	FF	LDAB ≠ FF
001E	5A		DECB
001F	27	F8	BEQ −8
0021	20	FB	BRA −5

(Continúa)

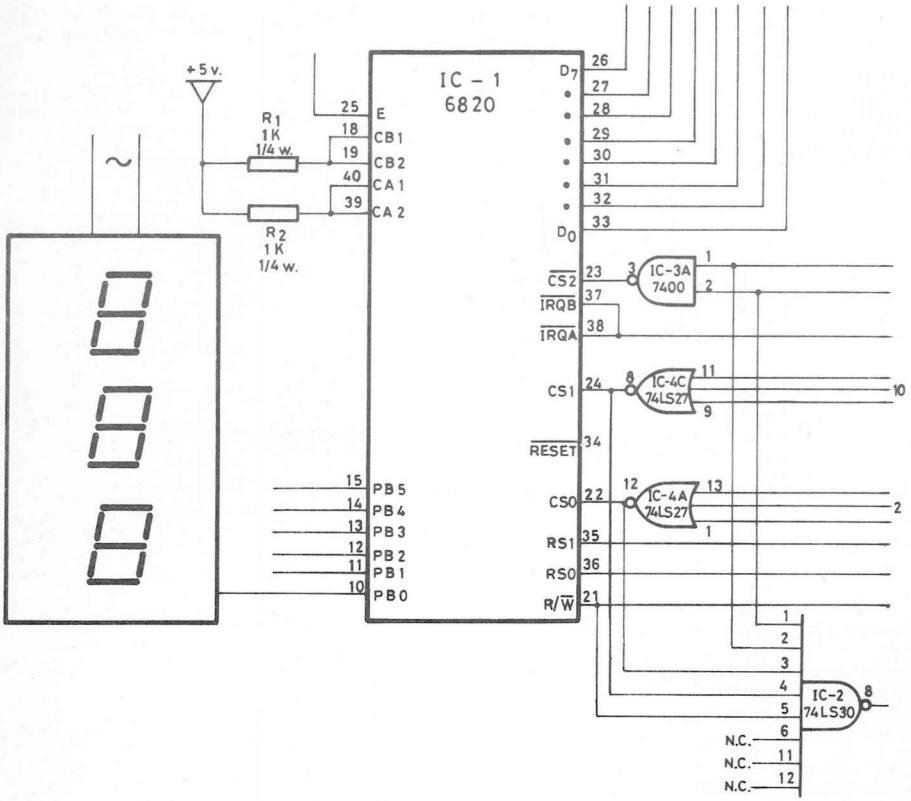


Fig. 8-45.–Conexión del PIA y la unidad contadora con el sistema.

(Continuación)

0023	86	FF	LDAA ≠ FF
0025	4A		DECA
0026	27	07	BEQ +7
0028	C6	FF	LDAB ≠ FF
002A	5A		DECB
002B	27	F8	BEQ −8
002D	20	FB	BRA −5
002F	7A	004d	DEC 004D
0032	96	4D	LDAA 4D
0034	27	02	BEQ +2
0036	20	DF	BRA −33
0038	86	01	LDAA ≠ 01
003A	B7	8000	STAA 8000

(Continúa)

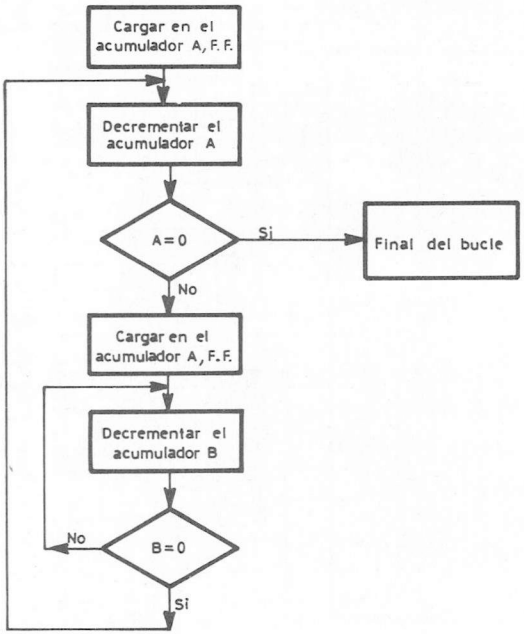


Fig. 8-46.–Ordiograma de un bucle apto para el microprocesador 6800.

(Continuación)

003D	7C	004C	INC 004C
0040	86	30	LDAA ≠ 30
0042	4A		DECA
0043	27	C9	BEQ –55
0045	C6	FF	LDAB ≠ FF
0047	5A		DECB
0048	27	F8	BEQ –8
004A	20	FB	BRA –5
004C	Posición del contador interno		
004D	Posición para variar el delay		

6.º PROYECTO: JUEGO PARA LA MEDIDA DE REFLEJOS

Este proyecto consiste en un juego, que permite medir la calidad o rapidez de los reflejos de una persona. El resultado se visualiza en una escala luminosa formada por 8 diodos luminiscentes tipo led. Dicha escala está graduada con 4 puntuaciones: excelente, bueno, regular y malo. Cuando se alcanza el rango de excelente, el sistema regala una nueva partida, siendo 3 el mayor número de partidas que pueden hacerse consecutivamente obteniendo la graduación de excelente.

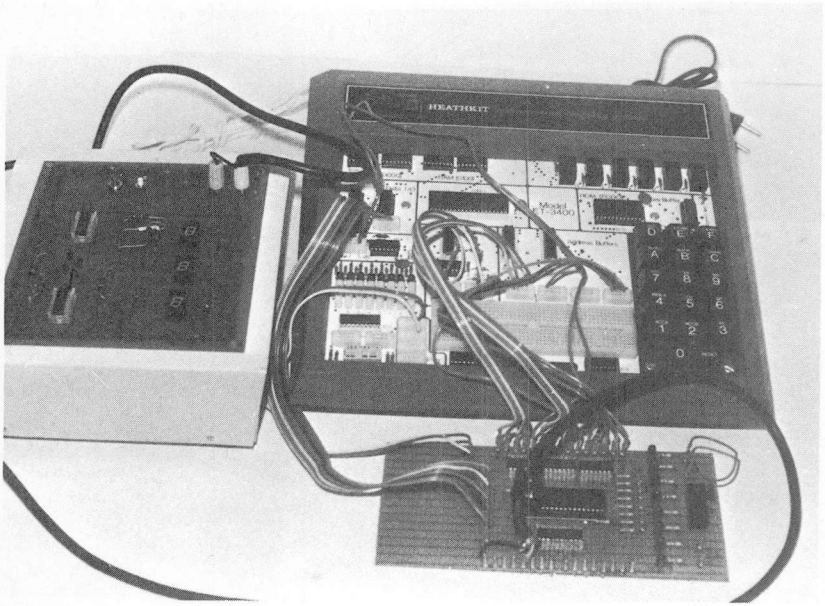


Fig. 8-47.–Foto de la unidad contadora adaptada a un microcomputador Heathkit, basado en el microprocesador Motorola 6800.

El sistema se inicia al pulsar una tecla, que sirve para poner en marcha el programa, el cual produce el encendido de un led cuando ha transcurrido un cierto tiempo (unos 10 segundos). El juego consiste en pulsar otro botón lo antes posible, después de haberse encendido el led de juego; según el tiempo que transcurra entre el encendido del led de juego y la pulsación de la tecla apropiada, se establece la calificación de los reflejos, siendo tanto peor cuanto más se tarde en apretar la tecla.

Solución con microprocesador R 6502

En la figura 8-48 se muestra una fotografía de la adaptación de la caja, que contiene los periféricos necesarios para el juego y el KIM-1.

La puerta A del R 6530 del KIM-1 actuará como salida. El bit de menos peso de la puerta B actúa como entrada de la información, que indica si el interruptor ha sido o no pulsado y, finalmente, los bits 1 y 2 de la puerta B quedan conectados como salidas a los leds de "partida" (cuando

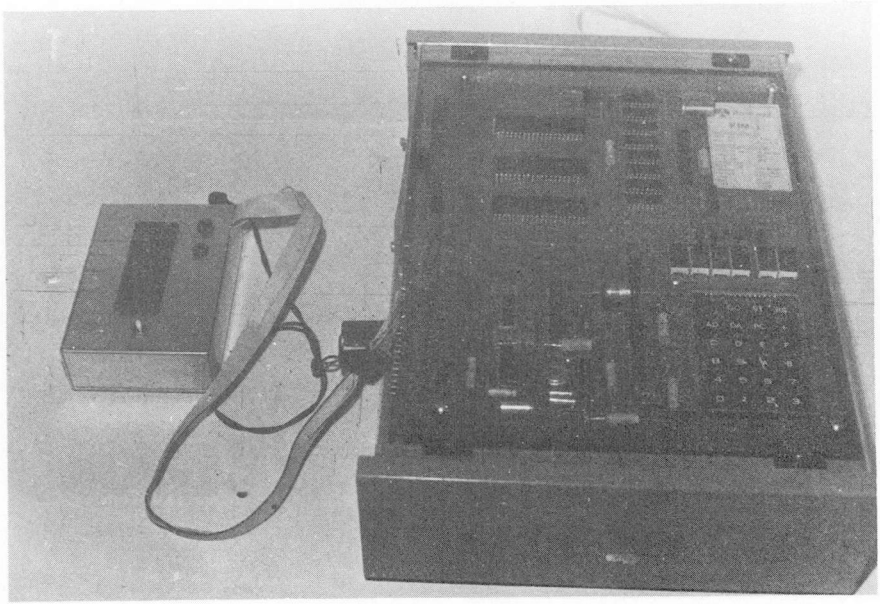


Fig. 8-48.– Foto del conexionado de los periféricos del juego de medida de reflejos con el KIM-1.

se alcanza "excelente") y de "juego" (que indica el comienzo del juego) respectivamente.

Los 8 bits de la puerta A quedan conectados a los 8 leds, que sirven para expresar la calificación obtenida.

En la figura 8-49 se muestra el circuito de conexión del pulsador, el cual está normalmen ' c(rado y, por lo tanto, aplicando un nivel 0 al pin PBO de la puerta B.

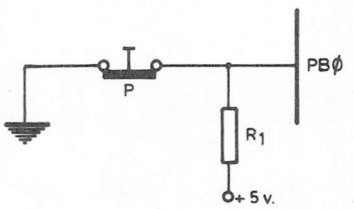


Fig. 8-49.–Circuito de conexión del pulsador al pin PBO.

La figura 8-50 muestra un detalle sobre el circuito del hardware auxiliar que rodea a los elementos periféricos.

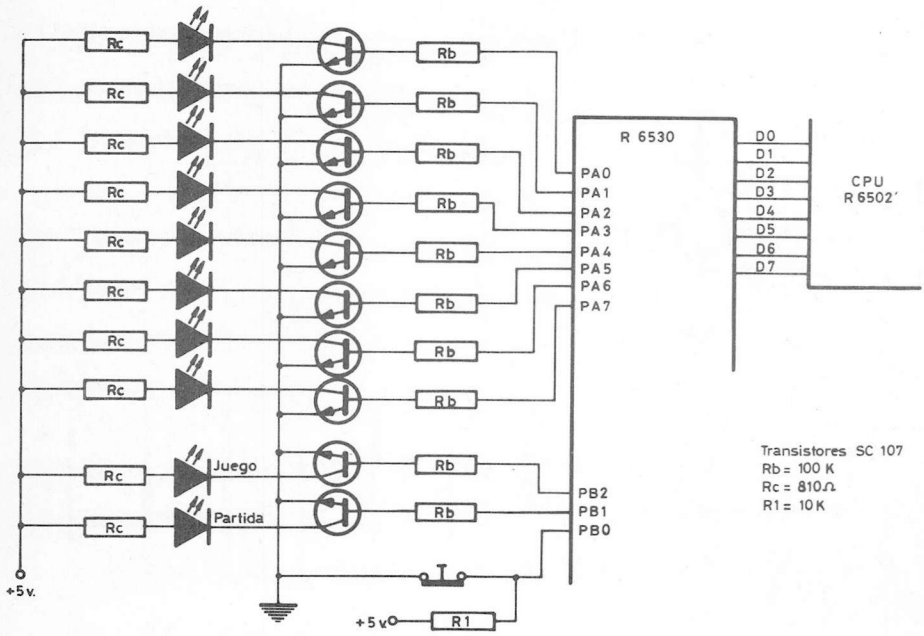


Fig. 8-50.–Conexión de los periféricos del juego de reflejos a los pines del R 6530.

En el caso de no pulsarse P, el primer bit del Acumulador se cargará con nivel 0, puesto que PBO valdrá 0. Al realizar la función AND del Acumulador con el valor 01, el flag Z queda a nivel 0, no cumpliéndose la condición de Z = 1, que se comprueba con la correspondiente instrucción de branch condicional.

Por el contrario, si P está pulsado, el Acumulador quedará cargado con el bit de menos peso a nivel 1 y la función AND con el valor 01 origina la puesta a 1 del flag Z.

Las salidas del R 6530 no tienen la suficiente potencia para excitar a un diodo led, por lo que hay que amplificarlas, por ejemplo, a través del circuito representado en la figura 8-51.

La graduación de los reflejos se realiza mediante 8 diodos leds, de forma que, cuando todos o todos menos uno están iluminados, se alcanza el nivel de "excelente". Cuando faltan por iluminar los 3 ó 4 superiores, se alcanza el nivel de "bueno". Si faltan 5 ó 6, el de "regular" y si faltan 7 ó los 8, el de "malo", tal como lo expresa la figura 8-52.

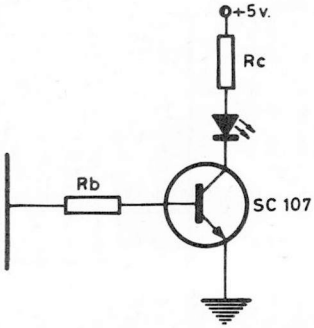


Fig. 8-51.–Excitación de un led desde un pin del R 6530 a través de un transistor.

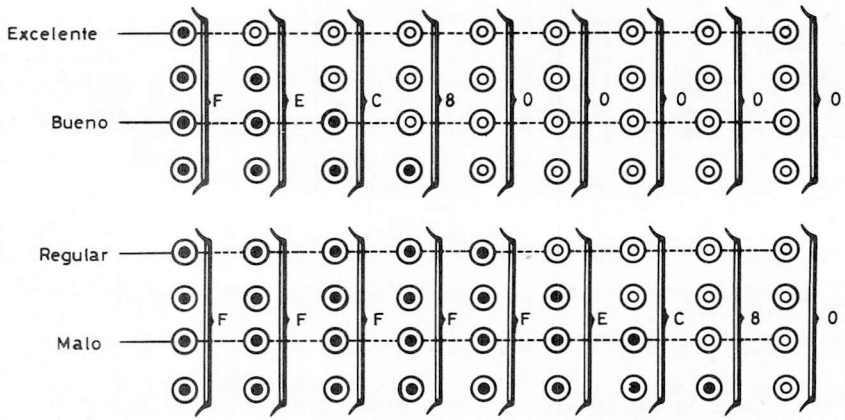


Fig. 8-52.–Calificaciones según el encendido de los leds.

Dependiendo de la rapidez de la pulsación de P, o del nivel de reflejos, se deberán encender una serie de leds según el siguiente orden:

Primeramente, en una posición de RAM (0002) cargamos el valor FF. Por cada vez que se pregunta si P está pulsado y sea que no, realizamos una función ASL con dicha posición. Figura 8-52 bis.

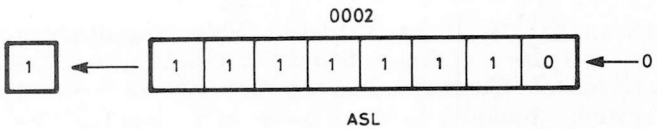


Fig. 8-52 bis.–Realización de la instrucción ASL.

Mientras P no se pulse, se seguirá haciendo la función ASL sobre la posición (0002) hasta que esta sea 00.

Cuando se pulsa P, el contenido de la posición 02 irá a parar a la escala de iluminación. Entre el testado de P y cada función ASL existe un pequeño delay.

En la figura 8-53 se muestra el ordinograma que cumple las condiciones de este juego.

Un breve comentario de los pasos del ordinograma es el siguiente:

1.º) Inicializar las puertas A y B del R 6530.

2.º) Delay grande, que decrementa FF veces el valor FF hasta anularlo.

3.º) Hay una pequeña tabla que se carga en el registro X y que indica el número de veces que se ha de repetir el delay anterior. Al hacer partida, se incrementa X y con ello se consigue que, en este caso, el delay varíe automáticamente.

4.º) Se enciende el led de "juego" y se debe pulsar lo antes posible P.

5.º) El sistema testa si se ha pulsado P y, si es así, saca el valor FF que ilumina todos los leds y marca la calificación de "excelente", caso contrario, se hace un pequeño delay y se testa la pulsación de P para, en caso afirmativo, sacar el siguiente valor FE y, en caso negativo, continuar la misma operación con todos los valores de la posición 2.

6.º) También hay que proceder al testado, cuando surge la calificación de "excelente", si es la 3.ª partida sucesiva la que se juega, por haber conseguido esta marca anteriormente. Si aún no se han hecho las 3 partidas, se procede a encender el led de "partida", visualizar la calificación de la partida gratis que se haya jugado y saltar a la dirección de inicio del programa (0205) para que con sólo pulsar la tecla de correr el programa (GO) se participe en otra partida. Dicho salto se realiza almacenando 05 en la dirección FA (POINT L), que corresponde a la posición en que se visualiza su contenido en los dos dígidos de menos peso del display de dirección del KIM-1, y 02 en la posición FB, cuyo contenido se visualiza mediante una rutina, ya explicada en los ejercicios iniciales de este capítulo, en los dos dígitos de más peso. La rutina de visualización comienza en la dirección 1C4F.

El contaje de las partidas se realiza desde una posición de memoria, que se incrementa cada vez que se enciende el led de partida. Dicha posición se carga en el Acumulador y se compara con 03 y, en caso de que dé resultado 0, indica que se han jugado las 3 partidas de premio.

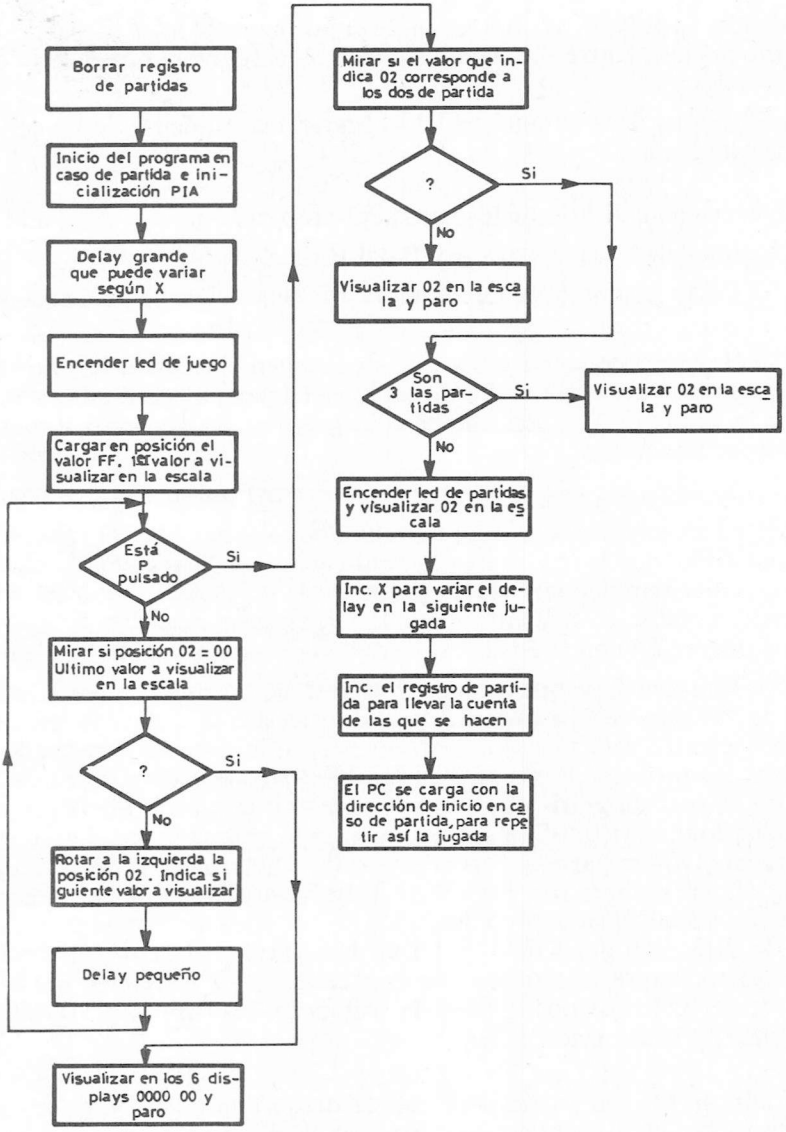


Fig. 8-53.– Ordinograma del juego de valoración de los reflejos.

El programa completo de este juego se muestra a continuación.

0200	A9 00	LDA ≠ 00	Borrar registro partidas.
0202	85 01	STA 01	
0204	A9 FF	LDA ≠ FF	
0206	8D 0117	STA 1701	Inicializar PIA.
0209	A9 06	LDA ≠ 06	
020B	8D 0317	STA 1703	
020e	A9 00	LDA ≠ 00	
0210	8D 0017	STA 1700	Borrar escala y leed de juego.
0213	8D 0217	STA 1702	
0216	20 8102	JSR 0281	A delay grande.
0219	A9 04	LDA ≠ 04	Encender leed de juego.
021B	8D 0217	STA 1702	
021E	A9 FF	LDA ≠ FF	Inicializar escala.
0220	85 02	STA 02	
0222	AD0217	LDA 1702	
0225	29 01	AND ≠ 21	Testar si P = 1 en cuyo caso vamos a la dirección 0250.
0227	F0 03	BEQ +3	
0229	4C 5002	JMP 0250	
022C	A5 02	LDA 02	Si 0002 = 0 resultado es "muy malo".
022E	F0 15	BEQ +21	
0230	A2 00	LDX ≠ 00	
0232	A9 02	LDA ≠ 02	
0234	95 00	STA z (pág. X)	Cargar posiciones 00 y 029B, así como el registro Y con valores para realizar el delay pequeño.
0236	A9 FF	LDA ≠ FF	
0238	8D 9B02	STA 029B	
023B	A0 FF	LDY ≠ FF	
023D	20 8C02	JSR 028C	A delay pequeño.
0240	06 02	ASL 02	Decrementar escala.
0242	4C 2202	JMP 0222	
0245	A9 00	LDA ≠ 00	
0247	85 F9	STA F9	
0249	85 FA	STA FA	Visualizar en los displays el valor 0000 00 (muy malo).
024B	85 FB	STA FB	
024D	4C 4F1C	JMP 1C4F	
0250	A5 02	LDA 02	
0252	C9 FF	CMP ≠ FF	Mirar si el contenido de 0002 corresponde a los dos valores de partida. Si no son esos valores, visualizar el contenido de 02 en la escala.
0254	F0 08	BEQ +8	
0256	C9 FE	CMP ≠ FE	
0258	F0 04	BEQ +4	
025A	8D 0017	STA 1700	
025D	00	BRK	

(Continúa)

(Continuación)

025E	A5 01	LDA 01	
0260	C9 03	CMP ≠ 03	Mirar si las partidas realizadas son 3.
0262	F0 18	BEQ +24	
0264	A9 02	LDA ≠ 02	
0266	8D 0217	STA 1702	Encender leed de partida y visualizar el
0269	A5 02	LDA 02	contenido de 02 en la escala.
026B	8D 0017	STA 1700	
026E	E8	INX	Variar el delay según X.
026F	E6 01	INC 01	
0271	A9 04	LDA ≠ 04	Incrementar el registro de partidas y car-
0273	85 FA	STA FA	gar el contador de programa con la direc-
0274	A9 02	LDA ≠ 02	ción donde empieza una nueva jugada.
0277	85 FB	STA FB	
0279	4C 4F1C	JMP 1C4F	
027C	A5 02	LDA 02	Si no hay partida, visualizar 02.
027E	4C 5A02	JMP 025A	
0281	D6 00	DEC z page	
0283	F0 15	BEQ +21	
0285	A9 35	LDA ≠ 35	Delay consistente en decrementar 35 ve-
0287	8D 9B02	STA 029B	ces el valor 35.
028A	A0 35	LDY ≠ 35	Este bucle se repetirá tantas veces como
028C	88	DEY	indique el contenido de la posición que
028D	F0 03	BEQ +3	direcciona el registro X.
028F	4C 8C02	JMP 028C	
0292	CE 9B02	DEC 029B	
0295	F0 EA	BEQ −22	
0297	4C 8A02	JMP 028A	
029A	60	RTS	

La figura 8-54 muestra una vista fotográfica de la caja sobre la que se han montado los diversos periféricos del juego. La figura 8-55 es el diagrama correspondiente a la grabación del circuito impreso, que se ha empleado para el montaje de los componentes.

Solución del 6.º proyecto con microprocesador Motorola 6800

El juego para la graduación de reflejos se ha diseñado de forma similar para el 6800, pero, al objeto de explicar algunas posibilidades en programación, en este caso se han variado principalmente los procesos de visualización de la siguiente manera:

1.º) El diseño a base del R 6502 dejaba todos los leds apagados y presentaba automáticamente en el display la dirección 0000.

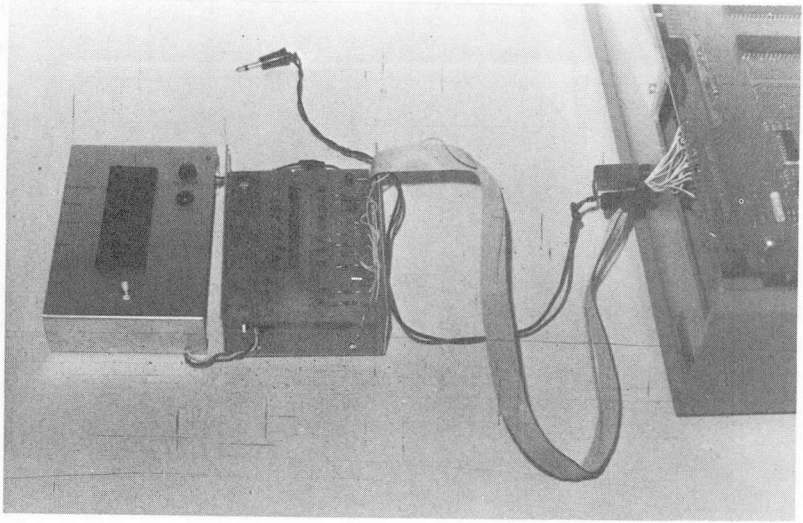


Fig. 8-54.–Fotografía de la caja que soporta los periféricos.

2.º) Cuando se ganaba partida con el R 6502, automáticamente el display presentaba la dirección de inicio de programa 0206 y, con sólo volver a pulsar GO, se iniciaba el juego.

Las subrutinas de visualización se han cambiado en el presente diseño de la siguiente forma:

1.º) Cuando, una vez encendido el led de juego, se tarda mucho en pulsar P, además de quedar apagados todos los leds de clasificación, aparecerá en el display el mensaje: MUY MALO.

2.º) Si la clasificación alcanzada no es suficiente para ganar partida, además de presentar los leds la graduación de los reflejos, el display presentará el mensaje METER DURO.

3.º) Cuando se gana partida, además de encenderse el led correspondiente, el display presenta el mensaje: "PULSAR RESET y D0 000B". La dirección 000B es la que corresponde a la inicialización del juego, en el caso de ganar partida. El número máximo de partidas gratis será de 3 y, aunque al término de la 3.ª se alcance "excelente", además de la graduación, el display presentará el mensaje METER DURO.

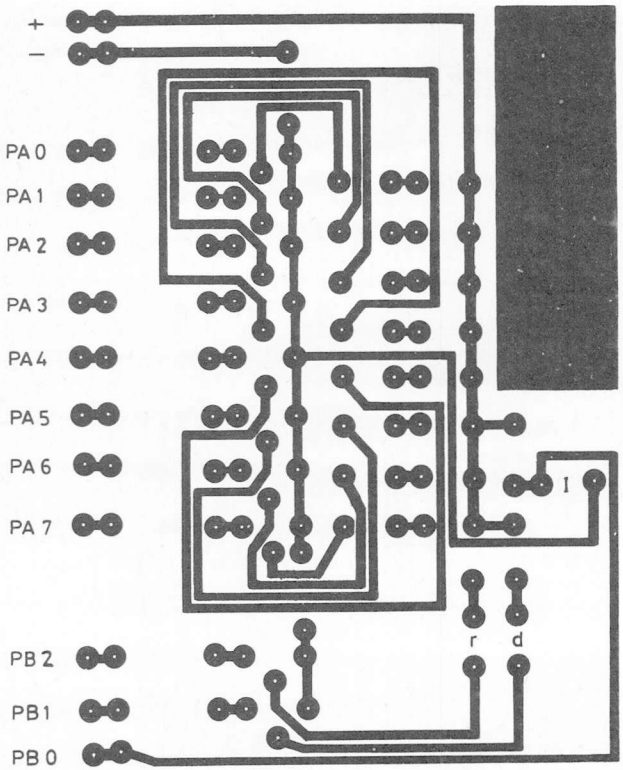


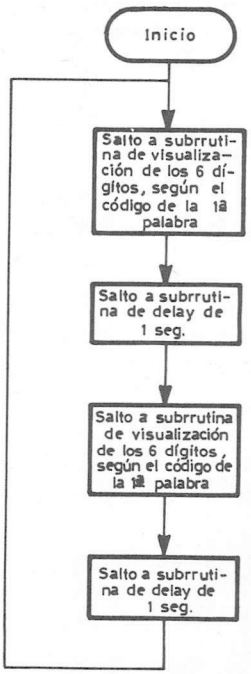
Fig. 8-55.—Dibujo del circuito impreso sobre el que se montan componentes.

En la figura 8-56 se muestra el ordinograma preciso para poder presentar en el display de 6 dígitos un mensaje de dos palabras.

En principio, para visualizar una palabra en los 6 dígitos del display, se necesita un código que determine qué dígitos y qué segmentos han de iluminarse. Teniendo formado este código, se pasa a una subrutina de visualización (cuyo inicio en el programa es la dirección FD8D). La subrutina toma el código y lo visualiza en los 6 dígitos. A continuación, se procede a pasar a otra subrutina de delay que temporiza 1 segundo, tiempo en el que permanecerá visualizada la palabra. Después, se repiten las mismas operaciones para visualizar la 2.ª palabra.

La formación de códigos consiste en determinar los segmentos de los dígitos, que se han de iluminar. Por ejemplo, para expresar el mensaje MUY MALO se comienza formando el código de la 1.ª palabra MUY,

Fig. 8-56.– Ordinograma para presentar dos palabras seguidas en los 6 dígitos del display.



que tiene el inconveniente que, al no poder formar la M con los segmentos, se ha procedido a sustituirla por una H. La figura 8-57 muestra el código necesario para la palabra MUY.

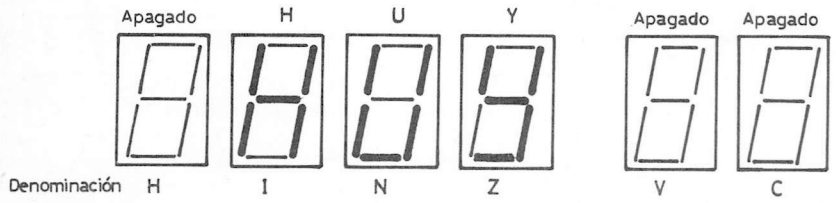


Fig. 54.–Código de formación de la palabra MUY. Los dígitos H, V y C deben quedar apagados.

La figura 8-58 muestra la denominación de los diferentes segmentos de uno de los dígitos del display.

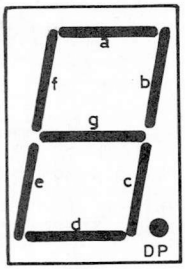


Fig. 8-58.–Denominación de los segmentos de un dígito.

La tabla de niveles que hay que aplicar a los segmentos para que todos quedasen apagados sería la mostrada en la figura 8-59.

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Segmento	Dp	a	b	c	d	e	f	g
Codigo	0	0	0	0	0	0	0	0
Hexadecimal	0				0			

Fig. 8-59.–Tabla de niveles para que queden apagados todos los segmentos.

El código necesario para formar la letra H, que se hace equivalente con la M, se muestra en la tabla de la figura 8-60.

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Segmento	Dp	a	b	c	d	e	f	g
Código H	0	0	1	1	0	1	1	1
Hexadecimal	3				7			

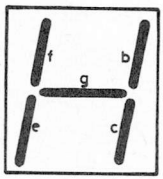
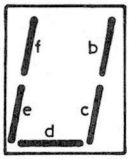


Fig. 8-60.–Código para formar la letra H, equivalente a M.

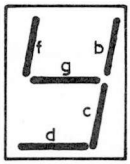
En la figura 8-61 se ofrecen las tablas de las restantes letras de la palabra MUY, así como una tabla resumen de los 6 dígitos para la palabra MUY y otro para la palabra MALO.

Cada palabra dispone por lo tanto de un código representado por una tabla de valores y cada una de ellas la presenta el display durante el delay de 1 segundo.

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Segmentos	Dp	a	b	c	d	e	f	g
Código U	0	0	1	1	1	1	1	0
Hexadecimal	3				E			



	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Segmentos	Dp	a	b	c	d	e	f	g
Código Y	0	0	1	1	1	0	1	1
Hexadecimal	3				B			



Dígito	H	I	N	Z	V	C
Estado	Apagado	H	U	Y	Apagado	Apagado
Código	00	37	3E	3B	00	00

Dígito	H	I	N	Z	V	C
Estado	Apagado	H	A	L	0	Apagado
Código	00	37	77	0E	7E	00

Fig. 8-61.–Tablas para formación de letras U e Y, así como tabla resumen para la palabra MUY y la palabra MALO. La M se hace equivalente a la H.

A continuación, se ofrece el listado correspondiente a la visualización del mensaje MUY MALO. Figura 8-62.

```
0100    BD FD8D     JSR FD8D        Salto a subrutina de visualización (1.ª palabra).
0103    00-37-3E-3B-00-00-80        Código para visualizar MUY.
010A    8D 0E       BRS +14         Salto a subrutina de delay.
010C    BD FD8D     JSR FD8D        Salto a subrutina de visualización (2.ª palabra).
010F    00-37-77-0E-7E-00-80        Código para visualizar MALO.
0116    8D 02       BRS +2          Salto a subrutina de delay.
0118    20 E6       BRA –26         Salto a volver a empezar.
011A    CE FF00     LDX ≠FF00
011D    09          DEX             Subrutina de delay.
011E    26 FD       BNE –3
0120    39          RTS             Retorno a subrutina.
```

Fig. 8-62.–Listado del programa para la visualización del mensaje MUY MALO.

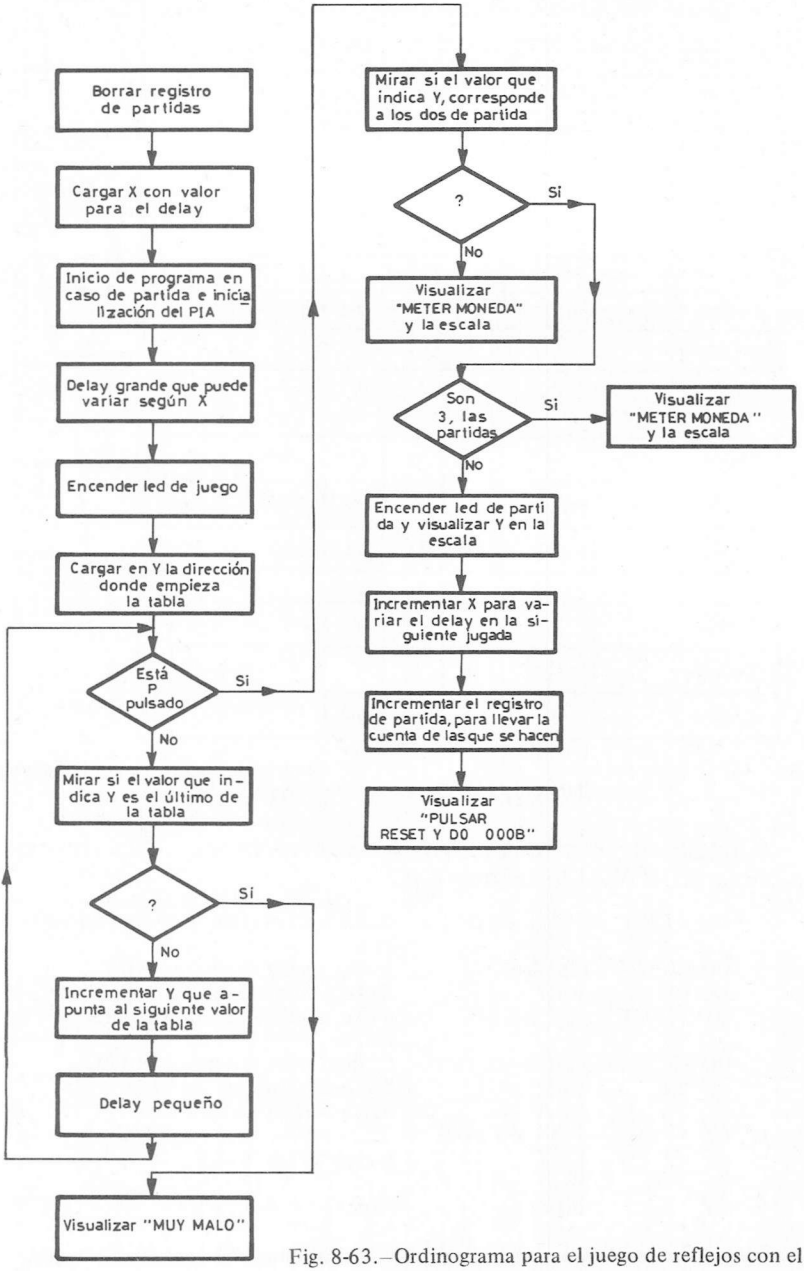


Fig. 8-63.—Ordinograma para el juego de reflejos con el 6800.

Los 3 mensajes que admite el juego se producirán de forma idéntica y en todos se usará la misma subrutina de delay.

Finalmente, se ofrece en la figura 8-63 el ordinograma de este diseño, en el que se puede apreciar la escasa diferencia con el empleado para el microprocesador R 6502.

Se presenta el programa correspondiente al ordinograma de la figura 8-63 con instrucciones del Motorola 6800.

0000	86 00	LDAA 00
0002	97 92	STAA 92 contador de programa
0004	CE0094	LDX con 1.ª tabla
0007	A6 00	LDAX indexadamente con el 1.º valor de la tabla
0009	97 93	STAA en 93 posición de delay variable
000B	86 FF	LDAA con FF
000D	B78000	STAA en la 8000
0010	86 04	LDAA con 04
0012	B78001	STAA en la 8001
0015	86 FE	LDAA con FE
0017	B78002	STAA en la 8002
001A	86 04	LDAA con 04
001C	B78003	STAA en la 8003
001F	86 FF	LDAA con FF
0021	C6 FF	LDAB con FF
0023	27 03	BEQ +3
0025	5A	DECB
0026	20 FB	BRA −5
0028	4A	DECA
0029	27 02	BEQ +2
002B	20 F4	BRA −12
002D	96 93	LDAA con la 93
002F	27 05	BEQ +5
0031	7A0093	DEC la 0093
0034	20 E9	BRA-23
0036	86 04	LDAA con 04
0038	B78002	STAA en la 8002
003B	CE0096	LDX con la 2.ª tabla
003E	F68002	LDAB con la 8002
0041	C4 01	ANDB 01
0043	C1 01	CMPB con 01
0045	27 1D	BEQ +29
0047	08	INX
0048	86 FF	LDAA con FF
004A	C6 20	LDAB con 20
004C	5A	DECB
004D	27 02	BEQ +2
004F	20 FB	BRA-5
0051	4A	DECA
0052	27 02	BEQ +2
0054	20 F4	BRA-12
0056	A6 00	LDAA indexadamente

(Continúa)

(Continuación)

```
0058    27 02       BEQ +2
005A    20 E2       BRA-30
005C    86 00       LDAA con 00
005E    B78000      STAA en la 8000
0061    7E0100      JMP al programa de visualización
0064    A6 00       LDAA indexadamente con la 2.ª tabla
0066    81 FF       CMPA con FF
0068    27 0A       BEQ +10
006A    81 FE       CMPA con FE
006C    27 06       BEQ +6
006E    B78000      STAA en la 8000
0071    7E0121      JUMP al programa de visualización
0074    D6 92       LDAB con la 92 contador de partidas
0076    C1 03       CMPB con 03
0078    27 16       BEQ +22
007A    C6 06       LDAB con 06
007C    F78002      STAB en la 8002
007F    B78000      STAA en la 8000
0082    7C0092      INC la 0092 contador de partidas
0085    CE0094      LDX con la 1.ª tabla
0088    08          INX
0089    A6 00       LDAA indexadamente
008B    97 93       STAA en la 93
008D    7E013D      JUMP programa de visualización
0090    20 DC       BRA -36
0092    XX          Contador de programa
0093    XX1         Posición de delay variable
0094    01-03       1.ª tabla
0096    FF-FE-FC-F8-F0-E0-C0-80-00 2.ª tabla

0100    BD FD 8D    JSR de visualización
0103    00-37-3E-3B-00-00-80 código a visualizar: "HUY HALO"
010A    8D 0E       BSR +15
010C    BD FD 8D    JSR de visualización
010F    00-3777-0E-7E-00-80 código a visualizar
0116    8D 02       BSR +2
0118    20 E6       BRA -26
011A    CE FF00     LDX con FF00
011D    09 26 FD    DEX y BNE - 3
0120    39          RTS

0121    BD FD 8D    JSR de visualización
0124    00-37-4F-07-4F-05-80 código a visualizar "METER"
012B    8D 0E       BSR +15
012D    BD FD 8D    JSR de visualización
0130    00-3D-3E-05-7E-00-80 código a visualizar "DURO"
0137    8D 02       BSR +2
0139    20 E6       BRA -26
013B    20 DD       BRA -35

013D    BD FD 8D    JSR de visualización
0140    67-3E-0E-5B-77-05-80 código a visualizar "PULSA r"
0147    8D 26       BSR +38
```

(Continúa)

(Continuación)

```
0149    BD FD 8D    JSR de visualización
014C    00-05-4F-5B-4F-07-80 código a visualizar "rESEt"
0153    8D 1A       BSR +26
0158    00-00-00-3B-00-00-80 código a visualizar "Y"
015F    8D 0E       BSR +15
0161    BD FD 8D    JSR de visualización
0164    3D-1D-7E-7E-7E-1F-80 código a visualizar "d0 000b"
016B    8D 02       BSR +2
016D    20 CE       BRA -50
016F    7E011A      JUMP a la 011A
```

La figura 8-64 muestra una fotografía de la adaptación del hardware exterior que compone los periféricos del juego con el sistema microcomputador, basado en el microprocesador de Motorola.

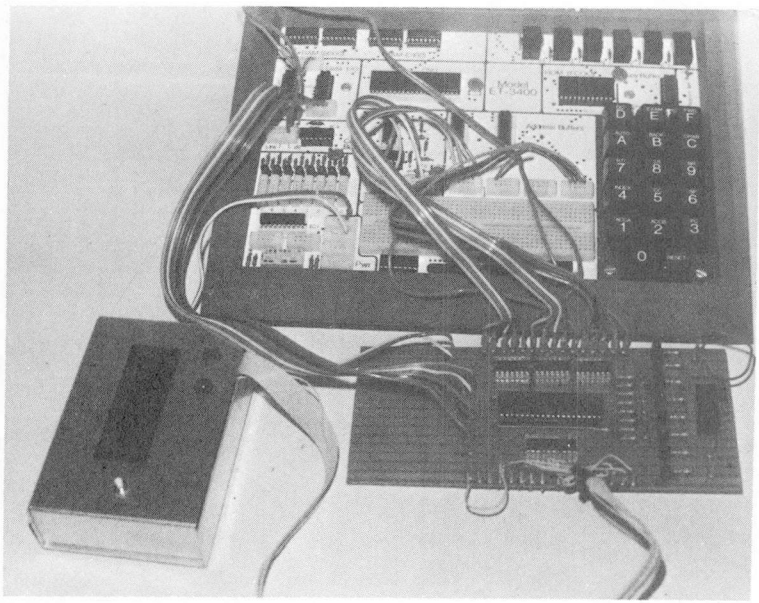


Fig. 8-64.—Adaptación de los periféricos del juego de medida de reflejos con el microcomputador, basado en el microprocesador de Motorola 6800.

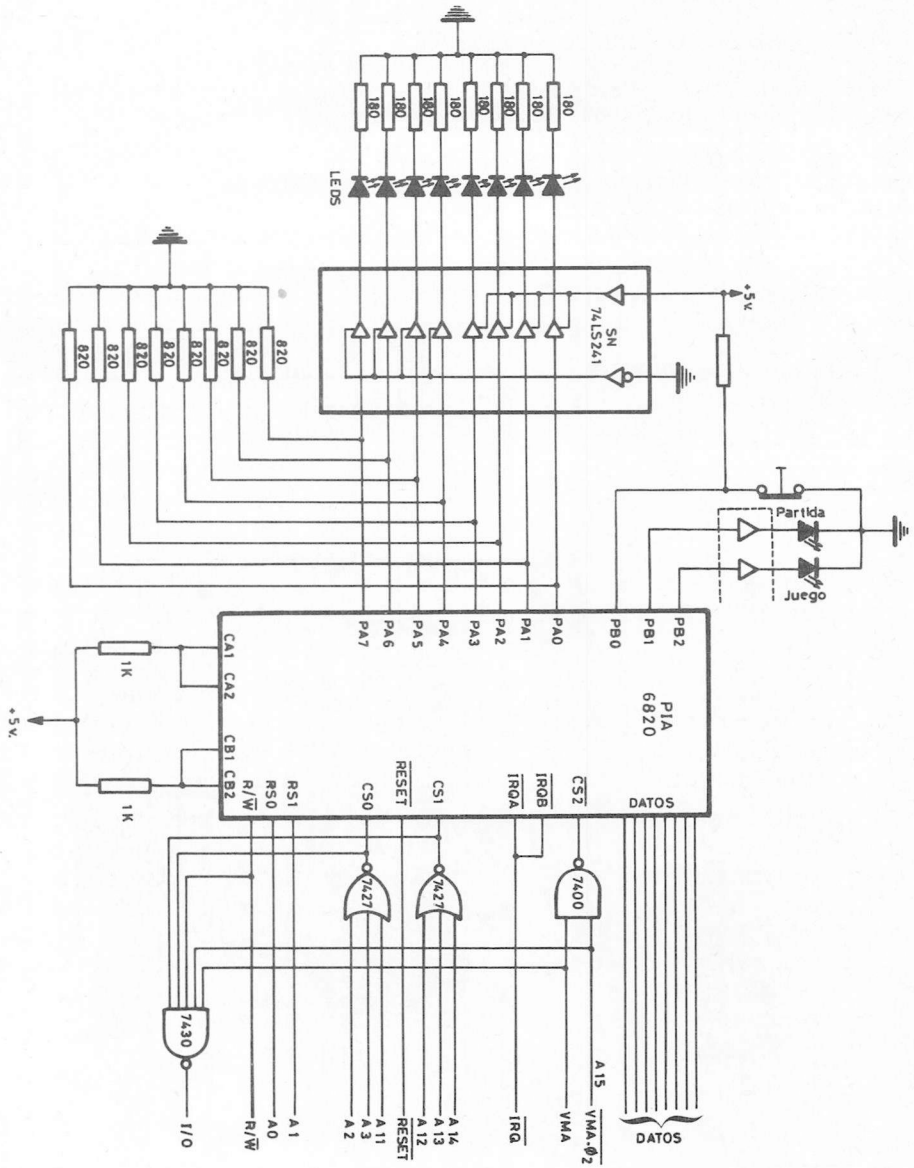


Fig. 8-65.–Diagrama de conexionado de los periféricos.

7.º PROYECTO: RESOLUCION DE ECUACIONES LOGICAS CON MICROPROCESADOR

Una de las aplicaciones más extendidas de los microprocesadores es la sustitución de circuitos lógicos a base de circuitos integrados SSI y MSI. Hay que tener en cuenta, sin embargo, que no en todos los casos es interesante sustituir un circuito lógico, montado con circuitos integrados de baja escala de integración, por un sistema con microprocesador. Es preciso que la complejidad del circuito lo haga propicio, así como el número de unidades a fabricar y los problemas que puedan derivarse de la velocidad, potencia consumida, cualificación de técnicos, diseño de placas de circuito impreso, presencia de ruidos e interferencias, adaptación a los periféricos, coste económico, fiabilidad, mano de obra empleada en el montaje, coste del software, previsión de futuros cambios en el funcionamiento, condiciones atmosféricas o ambientales de trabajo, facilidad para encontrar repuestos, mantenimiento y averías, alimentación, espacio ocupado y mil factores más, que han de ser estudiados meticulosamente antes de decidir la posibilidad de empleo de un sistema microprocesador. Por lo general, un primer factor a tener en cuenta es la complejidad, ya que, para diagramas lógicos en los que su implementación con circuitos integrados de baja escala de integración se requiera una cantidad que no sobrepase las 30 ó 40 unidades, hay que descartar el empleo del microprocesador.

Con objeto de dar una idea sobre la resolución de ecuaciones lógicas con microprocesador, se expone un sencillo ejemplo (para el cual no es aconsejable el uso del microprocesador en un caso real), que permita acometer problemas de mayor envergadura.

La idea básica consiste en emplear las 256 posiciones de la página cero, fácilmente direccionable, para contener las 256 combinaciones diferentes, que se pueden conseguir con 8 entradas, para controlar una salida.

En el ejemplo a resolver sólo se emplearán 3 entradas A, B y C, que pueden consistir en detectores, interruptores, captadores, etc. Como salida a controlar se dispondrá de un diodo luminiscente led, el cual deberá iluminarse en tres ocasiones:

1.ª) Si A no está activo, y B y C sí lo están.

2.ª) Si A está activo, y B y C no lo están.

3.ª) Si C está activo, y los otros dos no.

La tabla de la verdad, que responde al planteamiento del problema, se muestra en la figura 8-66.

C	B	A	L
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Combinaciones que activan el led

Fig. 8-66.–Tabla de la verdad del problema.

Tratando de resolver el problema mediante el microcomputador KIM-1, los 3 detectores, representados por simples interruptores, quedarán aplicados a los 3 bits de menos peso de la puerta A, la cual se programará como entrada (todos sus bits). Los 5 bits de más peso quedarán sin conexionar, por lo que siempre reflejarán el nivel 1. De esta forma se pueden obtener las 3 combinaciones de los 8 bits de la puerta A, que activarán a la salida, y que se representan en la tabla.

Cada una de las 3 combinaciones de la tabla responde a los códigos hexadecimales: F9, FC y FE. Ver figura 8-66 bis.

5 bits de más peso							
1	1	1	1	1	0	0	1
1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	0

F9
FC
FE

Fig. 8-66 bis.–Formación de códigos hexadecimales.

Los 3 valores que activan la salida se almacenarán en la página cero, añadiendo un cuarto valor cualquiera (pero diferente de los 3 anteriores), que sirve para indicar el final de la tabla. En este ejemplo usaremos el valor 33 como identificativo del final de la tabla en la página cero.

El ordinograma que resuelve el problema se basa en leer los valores de la tabla en página cero y compararlos con los que entran por la puerta A, que actúa de entrada. Cuando coincida un valor introducido por la puer-

ta A con uno de los de la tabla, se saca un nivel 1 por el pin PB0 de la puerta B, programado como salida para activar al led indicador.

Por supuesto que existen otros procedimientos menos laboriosos y más rápidos para resolver ecuaciones lógicas, pero el propuesto trata de generalizar el tema para poderlo aplicar a casos más complejos.

En la figura 8-67 se muestra el ordinograma de trabajo para resolver esta ecuación lógica, que puede traducirse como $L = \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot \overline{B} \cdot C$.

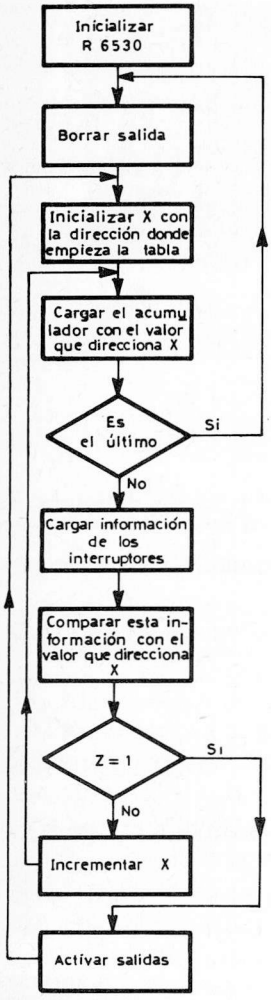
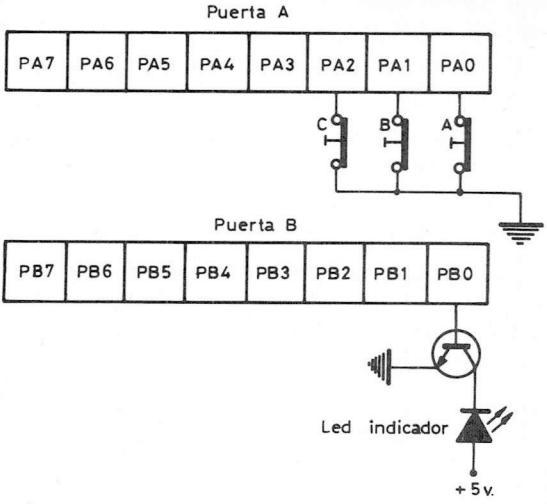


Fig. 8-67.–Ordinograma.

Fig. 8-68.–Diagrama simplificado de conexión de las entradas y salidas.



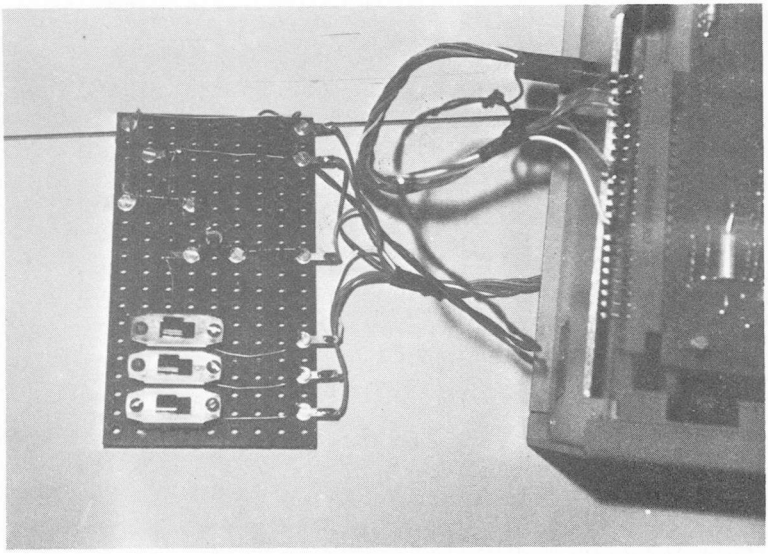


Fig. 8-68 Bis.–Foto del montaje que simula los periféricos de entrada y salida del 7º proyecto y su adaptación al KIM-1.

Como se aprecia en el ordinograma:

1.º) Se borra la salida para que quede apagado el led.

2.º) Se carga en el registro X, 00 que es la dirección de la página 0, en donde comienza la tabla de valores que activan la salida.

3.º) Se carga al Acumulador, mediante el direccionado "página cero, X".

4.º) Se compara el valor con 33 para determinar si ya se ha leído toda la tabla.

5.º) Si no se ha llegado al final de la tabla, se carga el Acumulador con la información de entrada de la puerta A y se compara con el valor que direcciona el registro X. Si la comparación es positiva, se saca un 1 por el pin de salida PB0 de la puerta B.

El listado completo del programa, que resuelve el ordinograma de la figura 8-67, es el siguiente:

```
0200    A9 01     LDA ≠ 01
0202    8D 0317   STA 1703      Inicializar la puerta A.
0205    A9 00     LDA ≠ 00
0207    8D 0117   STA 1701
```

(Continúa)

(Continuación)

020A	8D 0217	STA 1702	Borrar la salida. Puerta B.
020D	A2 00	LDX ≠ 00	Inicializar X.
020F	B5 00	LDA z (page, X)	
0211	C9 33	CMP ≠ 33	Mirar si es el último.
0213	F0 F0	BEQ −16	
0215	AD 0017	LDA 1700	Cargar entradas.
0218	D5 00	CMP z (page, X)	Mirar si lo cargado coincide con los valo-
021A	F0 04	BEQ +4	res que indica X.
021C	E8	INX	
021D	4C 0F02	JMP 020F	Volver a repetir todo lo anterior.
0220	A9 01	LDA ≠ 01	Sacar nivel "1" a la salida.
0222	8D 0217	STA 1702	
0225	4C 0D02	JMP 020D	

8.º PROYECTO: AUTOMATISMO RESUELTO CON MICROPROCESADORES

Aunque el presente ejemplo sobre resolución de un automatismo, empleando un sistema con microprocesador, no es aconsejable en la práctica, puesto que su sencillez recomienda resolverlo mediante circuitos integrados de baja escala de integración más simples y baratos, el objeto de exponerlo tiene la intención de aportar las ideas iniciales para solventar problemas de mayor envergadura.

Un carrito móvil puede desplazarse en dos direcciones, movido por un motor de c.c., que cuando se le invierte la polaridad cambia el sentido de giro y por tanto la dirección del carrito. La aplicación de la polaridad al motor se hace a través de dos relés, R_1 y R_2, que, según cuál se active, se determina la polaridad y el giro.

El carrito debe moverse entre dos finales de recorrido A y B, que pueden ser dos detectores de posición o dos pequeños microrruptores. Cada

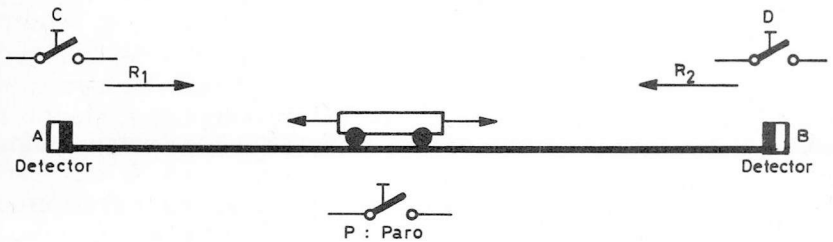


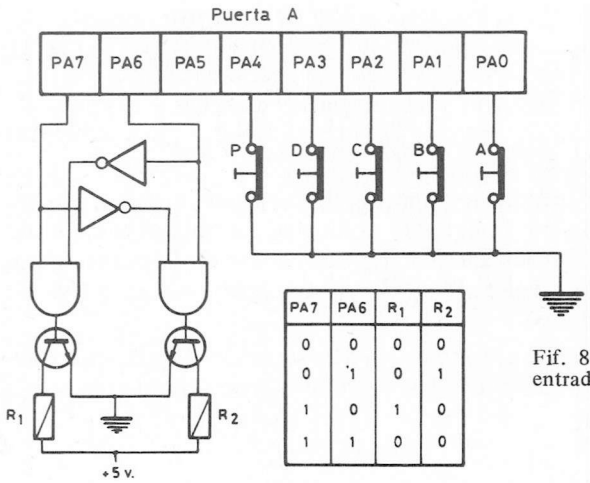
Fig. 8-69.–Esquema de funcionamiento del carrito móvil.

vez que el carro active uno de los finales de carrera, cambia automáticamente su dirección y se dirige hacia el otro detector. Es decir, que el carro rebota continuamente entre A y B.

Para gobernar los movimientos existen 3 pulsadores: el C, que activa a R_1 y mueve al carro hacia B, o sea, hacia la derecha en la figura 8-69 y el pulsador D que activa a R_2 y mueve al carro en sentido contrario. Finalmente hay un pulsador de paro, P, que permite detener al carro en cuanto se pulsa.

Solución con microprocesador R 6502

Empleando el microcomputador KIM-1, o un sistema similar, la puerta A del R 6530 actuará como elemento de entradas y salidas del sistema, siendo sus 2 bits de más peso las salidas que activarán los relés R_1 y R_2, mientras que sus 5 bits de menos peso recibirán las informaciones de los detectores o microrruptores A y B, así como de los pulsadores C, D y P.



PA7	PA6	R_1	R_2
0	0	0	0
0	1	0	1
1	0	1	0
1	1	0	0

Fif. 8-70.–Esquema simplificado de las entradas y salidas por la puerta A.

La figura 8-71 muestra la tabla de la verdad, que resuelve las combinaciones que activan los relés R_1 y R_2. Téngase en cuenta que el bit 6.° de la puerta A no se usa y, al estar desconectado, introduce siempre un nivel 1. Por otro lado los bits 7.° y 8.° interesa que, cuando se efectúan las entradas, estén siempre a nivel 0, para que se efectúe una operación AND entre el valor cargado al realizar la entrada de información y uno fijo que será 3F.

	5 bits de menos peso					Bits			Salidas	
	P	D	C	B	A	8º	7º	6º	R_1	R_2
Valor hexadecimal 21	0	0	0	0	1	0	0	1	1	0
" " 24	0	0	1	0	0	0	0	1	1	0
" " 22	0	0	0	1	0	0	0	1	0	1
" " 28	0	1	0	0	0	0	0	1	0	1
" " 30	1	0	0	0	0	0	0	1	0	0

Fig. 8-71.–Tabla de la verdad para la activación de los 2 relés.

Las funciones principales mostradas en el ordinograma de la figura 8-72, son las siguientes:

1.ª) Se inicializa puerta A (PIA) y se borran las salidas, para que el carro esté parado en principio.

2.ª) Se detecta si P está pulsado. Para ello se cargan las entradas en el Acumulador y se realiza una operación AND con el valor 10. Si está pulsado P, se borran las salidas.

3.ª) Si no está pulsado P, se inicializa X. Véase que, al final del programa ofrecido posteriormente, existen 2 valores que, sacados por la puerta A, el primero excita al relé R_1 y el segundo al R_2.

4.ª) Se pasa a testar el estado de los dos pulsadores C y D, así como el de los finales de carrera A y B. Si está activo A o C, se excita el relé R_1 sacando a la puerta A por el Acumulador el primer valor que direcciona el registro X. Si B o D se activan, se incrementa X y se saca por la puerta A, con lo que se excitará el relé R_2.

La detección del pulsador o microrruptor activado se logra, cargando en el acumulador el valor de las entradas y comparándolo con los de la tabla de la verdad mostrada anteriormente (21, 24, 22, 28).

El listado del programa correspondiente es:

```
0000    A9  C0     LDA ≠ C0
0002    8D  0117   STA 1701      Inicializar Puerta A.

0005    A9  00     LDA ≠ 00
0007    8D  0017   STA 1700      Borrar salida.

000A    AD  0017   LDA 1700      Cargar entradas.

000D    29  10     AND ≠ 10
000F    F0  03     BEQ +3
0011    4C  0500   JMP 0005      Mirar si está P pulsado.

0014    A2  3B     LDX 3B        Inicializar X.

0016    AD  0017   LDA 1700      Cargar entradas.
```

(Continúa)

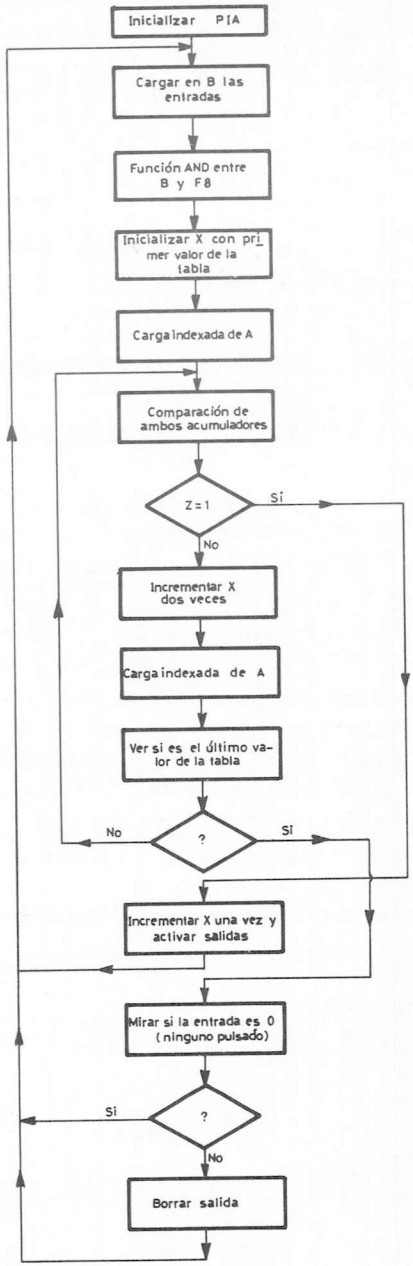


Fig. 8-72.– Ordinograma que resuelve el problema del automatismo del carrito.

(Continuación)

0019	29 3F	AND ≠ 3F	Poner los dos bits de más peso a "0".
001B 001D	C9 21 F0 14	CMP ≠ 21 BEQ +20	¿Está A activado?
001F 0021	C9 24 F0 10	CMP ≠ 24 BEQ +16	¿Está C activado?
0023 0025	C9 22 F0 0B	CMP ≠ 22 BEQ +11	¿Está B activado?
0027 0029	C9 28 F0 07	CMP ≠ 28 BEQ +7	¿Está D activado?
002B 002D	C9 20 F0 DB	CMP ≠ 20 BEQ −37	¿Hay alguno pulsado?
022F	4C 0500	JMP 0005	Borrar salidas.
0032	E8	INX	
0033 0035	B5 00 8D 0017	LDA z (page X) STA 1700	Activa salida correspondiente.
0038	4C 0A00	JMP 000A	Cargar entradas de nuevo.
003B 003C	80 40		Activar R_1 y R_2 respectivamente.

Solución con microprocesador Motorola 6800

La figura 8-73 muestra el conexionado de los detectores, pusladores y relés a la puerta A del PIA 6820, que adaptará las entradas y salidas al sistema microprocesador. Se han omitido las puertas lógicas de la figura 8-70, que evitan que se exciten los dos relés, al producirse un Reset que pone a 1 a todos los bits de la puerta A.

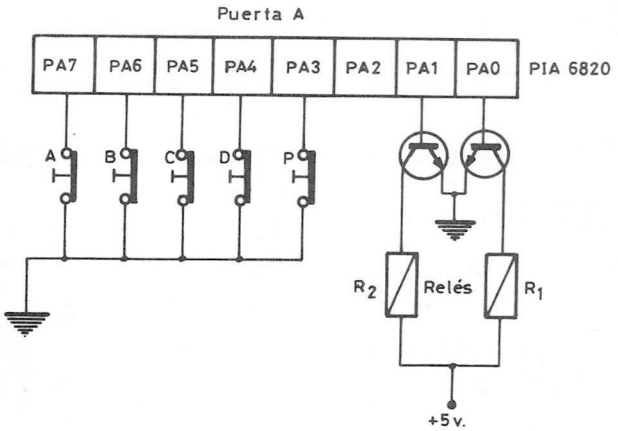


Fig. 8-73.–Diagrama simplificado de conexionado de las entradas y salidas del automatismo al PIA de adaptación al sistema.

A continuación se muestra la tabla de la verdad para la excitación de los 2 relés, teniendo en cuenta en este caso que, para obligar a los 3 bits de menos peso a valer siempre 0 cuando se realiza una entrada de información, se efectúa una operación AND de dicho dato de entrada con el valor F8.

		5 bits de más peso								Salidas	
		A	B	C	D	P				R_2	R_1
Valor hexadecimal	80	1	0	0	0	0	0	0	0	0	1
" "	40	0	1	0	0	0	0	0	0	1	0
" "	20	0	0	1	0	0	0	0	0	0	1
" "	10	0	0	0	1	0	0	0	0	1	0
" "	08	0	0	0	0	1	0	0	0	0	0

Fig. 8-74.–Tabla de la verdad, que define las combinaciones de los bits de la puerta A del PIA que activa a los relés R_1 y R_2.

La tabla de valores, que responde a las combinaciones de la tabla de la verdad precedidos por el valor CC, que identifica el final de la tabla, si se añade el valor de los 2 bits de salida, bien 01 si se activa R_1, ó 02 si se activa R_2, ó también 00 si no se activa ninguno de los dos, es la siguiente:

80-01, 40-02, 20-01, 10-02, 08-00, CC

Estos valores se almacenarán al final del programa, formando una tabla y se direccionarán mediante el registro X.

En la figura 8-75 se muestra el diagrama de flujo, en el que se destacan los siguientes pasos:

1.°) Inicializar PIA (posiciones 8000 y 8001) y cargar en Ac. B información de las entradas.

2.°) Operación AND con el valor F8 para poner a 0 los 3 bits de menos peso.

3.°) Se inicializa X con la dirección donde comienza la tabla.

4.°) Se carga el Ac. A de forma indexada con los valores de la tabla y se comparan con el contenido del Ac. B. Si la comparación es negativa $Z \neq 1$, se incrementa X en dos unidades, con lo que se pasa a testar el siguiente valor de la tabla y, si la comparación da resultado positivo ($Z = 1$), se incrementa X una sola vez, con lo que se sacan los niveles que excitan al relé correspondiente.

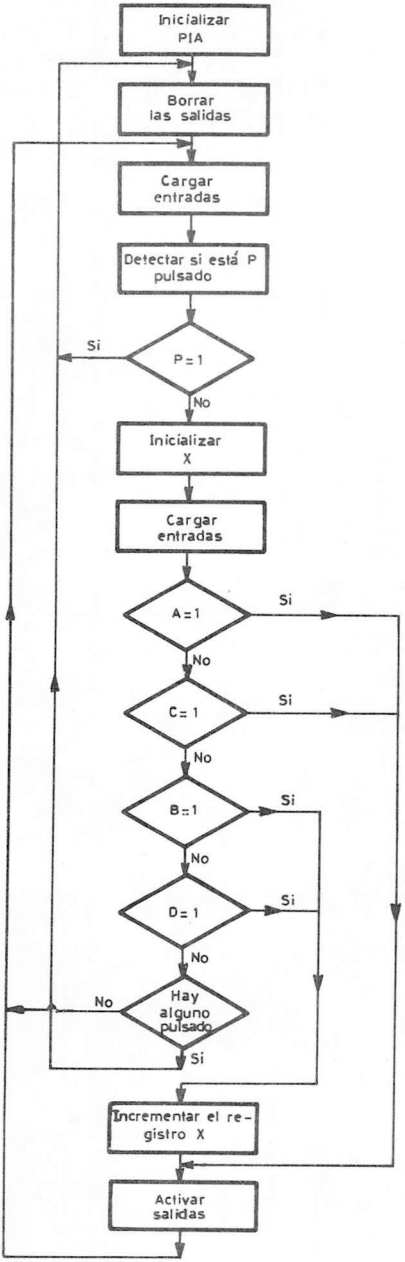


Fig. 8-75.–Ordinograma del automatismo con microprocesador 6800.

5.°) Existe una comparación con el valor CC, que indica el final de la tabla.

Finalmente, se ofrece el listado completo del programa, resuelto con el juego de instrucciones del microprocesador de Motorola 6800.

```
0000    86 03       LDAA ≠ 03
0002    B7 8000     STAA 8000
0005    86 04       LDAA ≠ 04
0007    B7 8001     STAA 8001
000A    F6 8000     LDAB 8000
000D    C4 F8       ANDB ≠ F8
000F    CE 0030     LDX ≠ 0030
0012    A6 00       LDAA index.
0014    11          CBA
0015    27 0A       BEQ +10
0017    08          INX
0018    08          INX
0019    A6 00       LDAA index.
001B    81 CC       CMPA ≠ CC
001D    27 0A       BEQ +10
001F    20 F3       BRA -13
0021    08          INX
0022    A6 00       LDAA index.
0024    B7 .8000    STAA 8000
0027    20 E1       BRA -31
0029    C1 00       CMPB ≠ 00
002B    27 FA       BEQ -6
002D    4F          CLRA
002E    20 F4       BRA -12
0030    80-01-40-02-20-01-10-02-08-00-CC
```

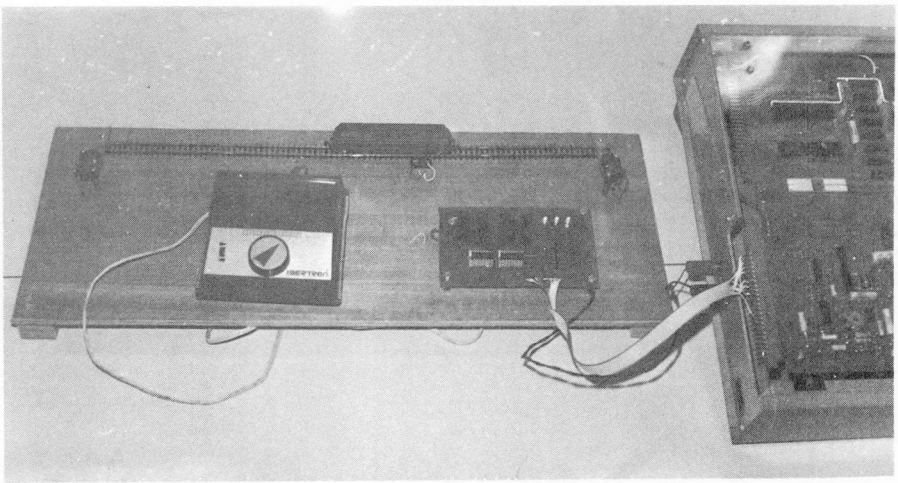


Fig. 8-76.–Fotografía del prototipo del 8° proyecto, en el que la máquina de un tren de juguete simula el movimiento del carrito.

La figura 8-76 muestra una fotografía del presente proyecto, en la cual se aprecia la simulación del carrito mediante una máquina de un tren eléctrico de juguete, el circuito de regulación de velocidad y la tarjeta de interface al KIM-1.

9.º PROYECTO: CONTROL DE APARCAMIENTO CON MICROPROCESADOR

Este proyecto, desarrollado mediante el microcomputador KIM-1, gobierna un aparcamiento de dos plantas, que acogen un total de 17 vehículos.

Un ascensor controlado por un motor de c.c., que cambia el sentido de giro al cambiar la polaridad que se le aplica, va colocando en la 1.ª planta hasta 7 coches y, cuando está llena, coloca otros 10 en la 2.ª planta. Completas las dos plantas, se impide el acceso al aparcamiento a otros coches.

En la iniciación del sistema se supone que el ascensor está situado en la planta baja, en donde está situado un semáforo que indica "paso prohibido" (rojo) a la espera de una información.

La detección o llegada de un coche al aparcamiento se advierte mediante el detector adecuado, simulado por el pulsador D. Esto provoca un testado de si existen plazas libres.

En el diseño, orientado a un fin exclusivamente didáctico, se han previsto 5 diodos leds indicadores. Tres de ellos, 2 rojos y uno verde, forman el semáforo de la entrada, de forma que, si se iluminan los 2 rojos, el verde debe quedar apagado, lo que significa "prohibido el paso". El verde encendido únicamente, indica "paso libre".

La restante pareja de leds (rojo y verde) indican tres funciones diferentes:

a) Si sólo está encendido el verde, significa que hay sitio en el aparcamiento para el coche detectado.

b) Caso de no haber sitio en el aparcamiento el led rojo se enciende simultáneamente con el verde.

c) Caso de iluminarse sólo el led rojo representa la realización de un programa de atención de interrupción, con el que se maneja manualmente el parking.

Si al detectarse un coche hay sitio, el semáforo se pondrá en verde, al igual que el led verde indicador de plazas libres. A continuación y después de un retraso de unos 30 segundos, que permiten colocarse en el as-

censor al coche, el semáforo pasa a rojo y el ascensor se pone en marcha hacia la primera planta con plazas libres, donde se detiene. Una vez sacado el coche de la plataforma del ascensor, se activa el pulsador C, situado en ambas plantas, con lo que el ascensor vuelve a la planta baja.

Para sacar un coche del aparcamiento se presionará el pulsador A o B, según el coche se encuentre en la 1.ª ó en la 2.ª planta. El ascensor se pone en movimiento hasta la planta desde la que se ha activado el pulsador y allí se detiene un tiempo de unos 30 segundos, que permiten la entrada del vehículo. Pasado este tiempo el ascensor baja hasta la entrada por donde puede salir el coche.

El semáforo permanecerá en rojo siempre que el ascensor esté en marcha o el sistema espere órdenes.

Debe tenerse muy presente que el sistema lleva la cuenta de los coches que entran y salen, y la inicialización del mismo representa el borrado de dicho contaje.

La complejidad del proyecto es debida al aprovechamiento de un juguete para su simulación y al intento de no elevar el número de entradas y salidas que dispone el KIM-1. La figura 8-77 muestra una fotografía del proyecto simulado en un juguete basado en un garage de dos plantas.

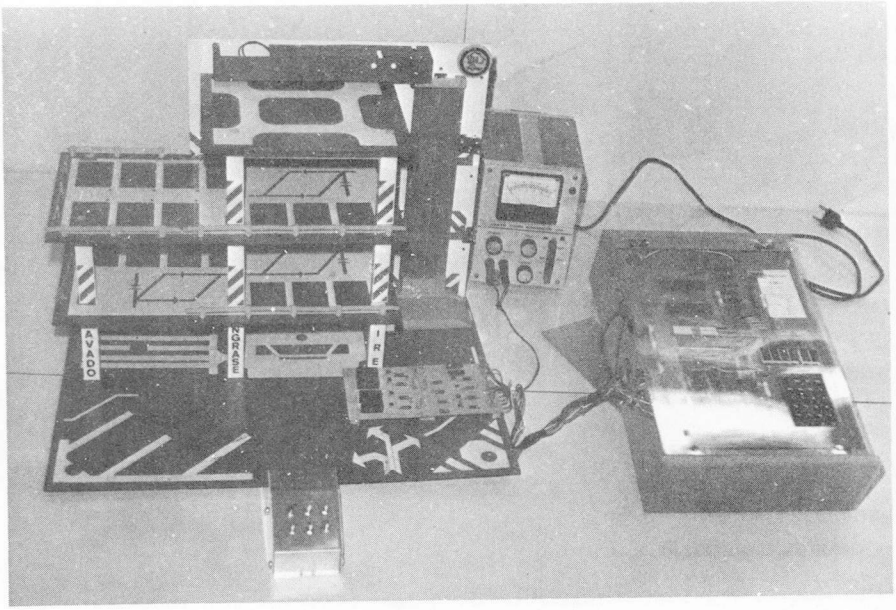


Fig. 8-77.–Simulación del proyecto sobre un garage de juguete.

"Control manual": Para resolver los casos, en los que fallen detectores o pulsadores, corte de alguna línea de entrada de información, defectos en la colocación del coche en la plataforma, etc., se ha previsto un control manual del sistema. Dicho control queda establecido cuando se produce una interrupción IRQ al presionar el pulsador de "control manual" situado en el pupitre de mandos. Cuando se produce esta interrupción, el sistema deja de obedecer a los detectores y pulsadores del "control automático" y sólo reconoce las órdenes que proceden del pupitre, que pueden ser de subida, bajada y paro del ascensor.

El programa de atención a la interrupción de "control manual" comienza en la dirección 0309, que como es sabido deberá cargarse en las posiciones correspondientes al vector de interrupción IRQ.

Reparada la avería que ha originado el empleo del "control manual", para retornar al control automático basta pulsar la tecla con este nombre que hay en el pupitre, con lo que se vuelve al programa principal.

La interrupción IRQ comentada, no afecta a los contadores de los coches que entran y salen, pero éstos dejan de seguir llevando la cuenta durante el "control manual". Al retornar al programa principal, quedan cargados los contadores con la misma cantidad que cuando salieron del mismo.

Al producirse la interrupción IRQ, se detiene el ascensor, se activa el led de ocupado y se apaga el de libre, poniendo con carácter permanente el semáforo en rojo.

Se presenta seguidamente una relación de los símbolos empleados con los diferentes detectores, pulsadores e indicadores, así como en la figura

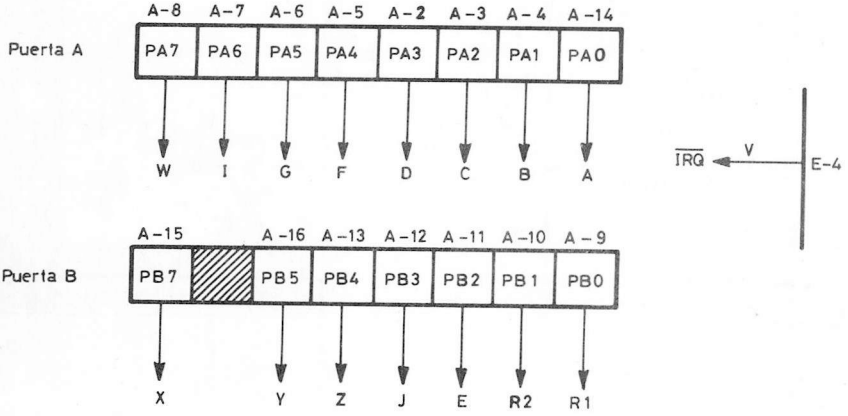


Fig. 8-78.—Asignación de las entradas y salidas del sistema a las puertas A y B del KIM-1.

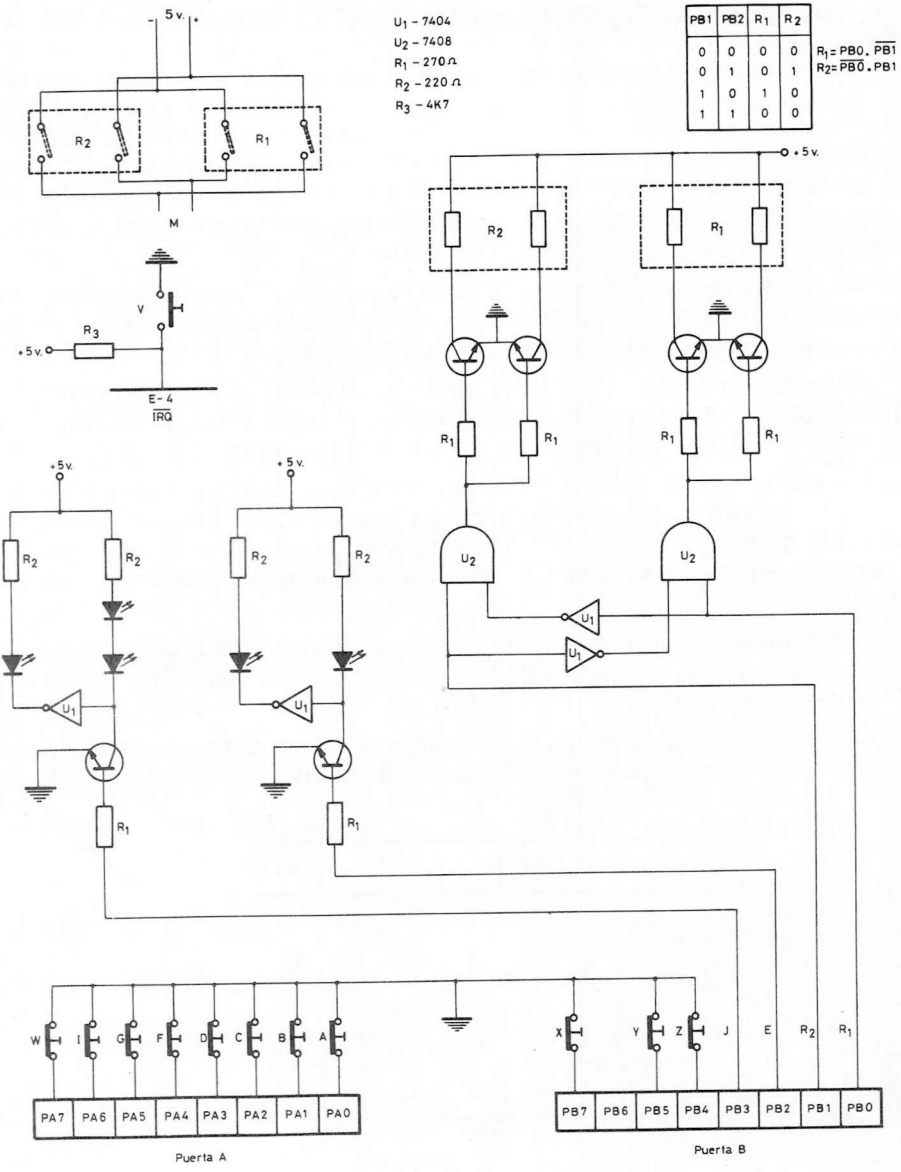


Fig. 8-79.–Esquema electrónico del hardware auxiliar que se ha de colocar externamente para la adaptación del sistema desarrollado a las puertas de entrada y salida del KIM 1.

8-78 la estructura y asignación de las entradas y salidas del sistema a las puertas A y B del microcomputador KIM-1.

FUNCION Y SITUACION DE LOS DETECTORES Y PULSADORES

A. Pulsador de llamada de la 1.ª planta.
B. Pulsador de llamada de la 2.ª planta.
C. Pulsador de bajada para cualquiera de las dos plantas.
D. Detector de presencia de coche para subir.
F. Detector de presencia del ascensor en la 1.ª planta.
G. Detector de presencia del ascensor en la 2.ª planta.
I. Detector de presencia del ascensor en la planta baja.
E. Leed rojo indicador de "PARKING COMPLETO".
$\overline{E}$. Leed verde indicador de "PARKING LIBRE".
J. Leeds (2) rojos del semáforo.
$\overline{J}$. Leed verde del semáforo.
R1. Salida a R1 (el ascensor sube).
R2. Salida a R2 (el ascensor baja).

V. Petición de interrupción IRQ.
W. Retorno o fin de interrupción.
X. Sube el ascensor.
Y. Baja el ascensor.
Z. Para el ascensor.

Estos pulsadores están localizados en el pupitre de mando y únicamente actúan cuando se produce la interrupción.

En la figura 8-79 se muestra el esquema electrónico de adaptación del KIM-1 al sistema. En la figura 8-80 se ve una fotografía del aparcamiento, sobresaliendo la placa sobre la que se ha implementado el hardware externo y el pupitre de mandos.

En la figura 8-81 se ofrece el ordinograma fundamental, que se ha seguido para desarrollar el programa que controla al sistema proyectado.

PROGRAMA PARA EL PROYECTO DEL APARCAMIENTO

Dirección	Código OP	Nemónico	Comentarios
0200	A9 00	LDA ≠ 00	
0202	85 01	STA 01	Borrar registros.
0204	85 02	STA 02	
0206	58	CLI	
0207	8D 0117	STA 1701	
020A	A9 0F	LDA ≠ 0F	Inicializar Puertas A y B.
020C	8D 0317	STA 1703	

(Continúa)

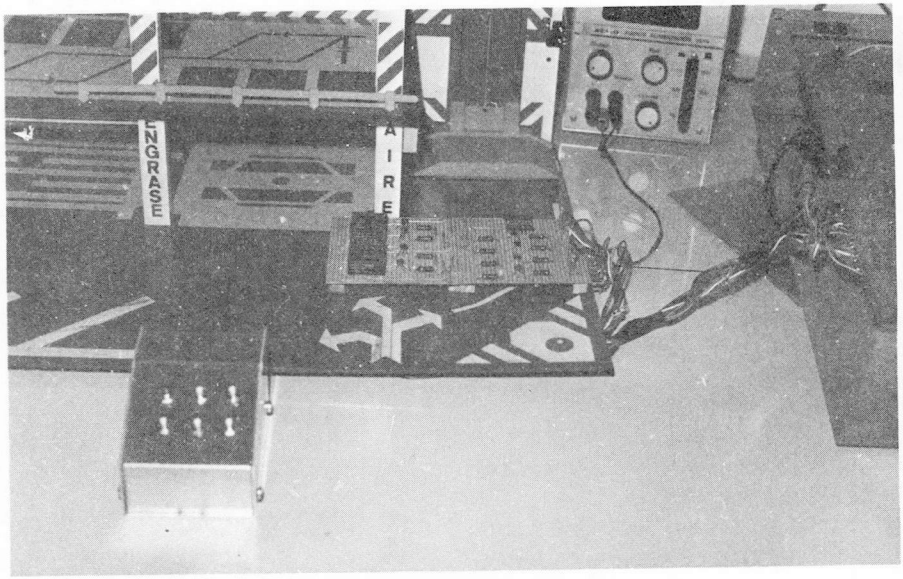


Fig. 8-80.–Foto de la placa con el hardware exterior y el pupitre de mandos.

(Continuación)

020F	A9 08	LDA ≠ 08	Situación de paro y semáforo en rojo.
0211	9D 0217	STA 1702	
0214	AD 0017	LDA 1700	
0217	85 00	STA 00	Testar si A = 1 (llaman de 1.ª planta), en
0219	29 01	AND ≠ 01	cuyo caso sube el ascensor.
021B	F0 03	BEQ +3	
021D	4C 6602	JMP 0266	
0210	A5 00	LDA 00	
0222	29 02	AND ≠ 02	Testar si B = 1 (llaman de 2.ª planta), en
0224	F0 03	BEQ +3	cuyo caso sube.
0226	4C A502	JMP 02A5	
0229	A5 00	LDA 00	
022B	29 08	AND ≠ 08	Testar si D = 1 (si hay coche para subir).
022D	F0 D7	BEQ −41	Si no hay, volver a inicio.
022F	A5 01	LDA 01	¡COCHE! Mirar si está libre 1.ª planta,
0231	C9 07	CMP ≠ 07	en cuyo caso se pasa a la dirección 0246.
0233	D0 11	BNE +17	Si está libre, se testa la 2.ª.
0235	A5 02	LDA 02	Mirar si está libre 2.ª planta, en cuyo caso
0237	C9 0A	CMP ≠ 0A	se pasa a la dirección 0285. Si no está, se
0239	D0 08	BNE +8	activa E (ocupado).

(Continúa)

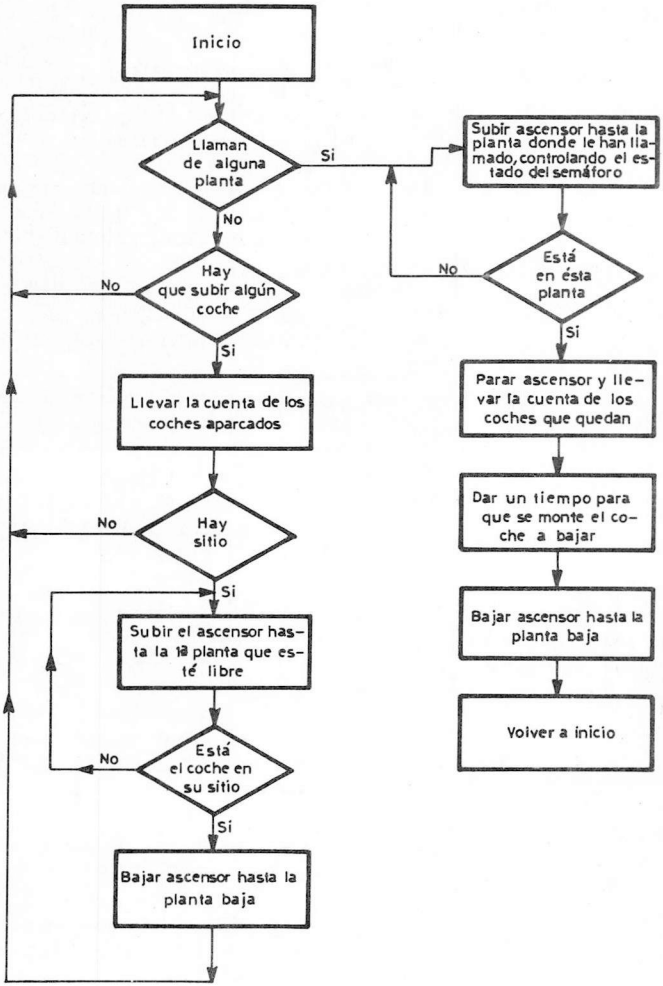


Fig. 8-81.–Ordinograma general del proyecto del aparcamiento.

(Continuación)

023B	A9	0C	LDA ≠ 0C	Activar E (ocupado), activar semáforo (rojo) y volver a inicio.
023D	8D	0217	STA 1702	
0240	4C	0602	JMP 0206	
0243	4C	8502	JMP 0285	
0246	A9	00	LDA ≠ 00	1.ª planta libre; desactivar E (libre); Poner semáforo en verde.
0248	8D	0217	STA 1702	

(Continúa)

(Continuación)

024B	20 E202	JSR 02E2	A subrutina de delay.
024E	E6 01	INC 01	Incrementar 1.er contador. Semáforo en rojo y R1 activo (sube).
0250	A9 09	LDA ≠ 09	
0252	8D 0217	STA 1702	
0255	AD 0017	LDA 1700	Testar si F = 1 (detector de presencia en 1.ª planta). Si F = 0, R1 sigue activo, y si F = 1 se pasa a subrutina de bajada.
0258	29 10	AND ≠ 10	
025A	F0 F4	BEQ −12	
025C	20 C402	JSR 02C4	A BAJADA.
025F	AD 0017	LDA 1700	Testar si A = 1 (llaman de 1.ª planta). Si no volver al inicio. Si A = 1 subir.
0262	29 01	AND ≠ 01	
0264	F0 DA	BEQ −38	
0266	C6 01	DEC 01	Decrementar 1.er contador, semáforo rojo y R1 activo (sube).
0268	A9 09	LDA ≠ 09	
026A	8D 0217	STA 1702	
026D	AD 0017	LDA 1700	Testar, si F = 1 (presencia en 1.ª planta); si no es, sigue activado R1. Si F = 1, para el motor (desactivar R1 y R2) manteniendo el semáforo en rojo.
0270	29 10	AND ≠ 10	
0272	F0 F4	BEQ ≠ 12	
0274	A9 08	LDA ≠ 08	
0276	8D 0217	STA 1702	
0279	20 E202	JSR 02E2	A DELAY.
027C	20 D002	JSR 02D0	A BAJADA.
027F	20 E202	JSR 02E2	A DELAY.
0282	4C 0602	JMP 0206	A INICIO.
0285	A9 00	LDA ≠ 00	2.ª planta libre; semáforo en verde y E desactivado (libre).
0287	8D 0217	STA 1702	
028A	20 E202	JSR 02E2	A DELAY.
028D	E6 02	INC 02	Incrementar 2.º contador, semáforo en rojo y R1 activo (sube).
028F	A9 09	LDA ≠ 09	
0291	8D 0217	STA 1702	
0294	AD 0017	LDA 1700	Testar, si G = 1 (presencia en 2.ª planta) si no es, R1 activo y si es pasa a bajada.
0297	29 20	AND ≠ 20	
0299	F0 F4	BEQ −12	
029B	20 C402	JSR 02C4	A BAJADA.
029E	AD 0017	LDA 1700	Testar, si B = 1 (llaman de 2.ª planta). Si no pasa a inicio. Si es sí subir (R1 activo).
02A1	29 02	AND ≠ 02	
02A3	F0 DD	BEQ −35	
02A5	C6 02	DEC 02	Decrementar 2.º contador, semáforo en rojo y R1 activo (sube).
02A7	A9 09	LDA ≠ 09	
02A9	8D 0217	STA 1702	

(Continúa)

(Continuación)

```
02AC  AD 0017  LDA 1700
02AF  29 20    AND # 20      Testar, si G = 1 (presencia en 2.ª planta)
02B1  F0 F4    BEQ -12       si no es sigue activado R1. Si es, para el
02B3  A9 08    LDA # 08      motor manteniendo el semáforo en rojo.
02B5  8D 0217  STA 1702
02B8  20 E202  JSR 02E2      A DELAY.
02BB  20 D002  JSR 02D0      A BAJADA.
02BE  20 E202  JSR 02E2      A DELAY.
02C1  4C 0602  JMP 0206      A INICIO.
```

SUBRUTINA DE BAJADA

```
02C4  A9 08    LDA # 08      Parar motor, semáforo en rojo.
02C6  8D 0217  STA 1702
02C9  AD 0017  LDA 1700      Testar, si C = 1 (bajada para las dos plan-
02CC  29 04    AND # 04      tas) si no es se mantiene el motor en paro.
02CE  F0 F4    BEQ -12
02D0  A9 0A    LDA # 0A
02D2  8D 0217  STA 1702      Si C = 1, R2 (baja) activo hasta que I = 1
02D5  AD 0017  LDA 1700      (presencia en planta baja).
02D8  29 40    AND # 40
02DA  F0 F4    BEQ -12
02DC  A9 08    LDA # 08      I = 1, parar motor, semáforo en rojo y re-
02DE  8D 0217  STA 1702      torno.
02E1  60       RTS
```

SUBRUTINA DE DELAY

```
02E2  A9 26    LDA # 26
02E4  85 03    STA 03
02E6  A9 30    LDA # 30
02E8  85 04    STA 04        Este delay de aproximadamente 30 seg.
02EA  A2 30    LDX # 30      consiste en decrementar 30 veces el valor
02EC  A0 30    LDY # 30      30. Ambos valores están contenidos en
02EE  88       DEY           los registros X e Y.
02EF  F0 03    BEQ +3        El bucle previo se repite 30 veces, según
02F1  4C EE02  JMP 02EE      la posición de memoria 0004.
02F4  CA       DEX           Todo lo anterior se repite tantas veces co-
02F5  F0 03    BEQ +3        mo lo indique el contenido de la posición
02F7  4C EC02  JMP 02EC      0003, en la cual se carga el valor 26 para
02FA  C6 04    DEC 04        conseguir el delay deseado de unos 30 seg.
02FC  F0 03    BEQ +3
02FE  4C EA02  JMP 02EA
0301  C6 03    DEC 03
0303  F0 03    BEQ +3
0305  4C E602  JMP 02E6
0308  60       RTS
```

(Continúa)

(Continuación) **INTERRUPCION IRQ**

0309	A9	0C	LDA ≠ 0C	Para motor, semáforo en rojo y E activo ocupado".
030B	8D	0217	STA 1702	
030E	AD	0017	LDA 1700	Testar, si W = 1 (retorno de interrupción). Si W = 1, se retorna; si no se mantiene bajo control manual.
0311	29	80	AND ≠ 80	
0313	F0	03	BEQ +3	
0315	4C	4003	JMP 0340	
0318	AD	0217	LDA 1702	
031B	85	00	STA 00	
031D	29	10	AND ≠ 10	Testar, si Z = 1 (paro).
031F	F0	03	BEQ +3	
0321	4C	0903	JMP 0309	
0324	A5	00	LDA 00	Testar, si X = 1; en cuyo caso sube el ascensor.
0326	29	80	AND ≠ 80	
0328	F0	08	BEQ +8	
032A	A9	0D	LDA ≠ 0D	X = 1, lo que activa R1 (sube), manteniendo el semáforo en rojo y E activo.
032C	8D	0217	STA 1702	
032F	4C	0E03	JMP 030E	
0332	A5	00	LDA 00	Testar, si Y = 1, en cuyo caso se baja.
0334	29	20	AND ≠ 20	
0336	F0	D6	BEQ −42	
0338	A9	0E	LDA ≠ 0E	Y = 1, lo que activa R2 (baja), manteniendo el semáforo en rojo y E activo.
033A	8D	0217	STA 1702	
033D	4C	0E03	JMP 030E	
0340	A2	FF	LDX ≠ FF	
0342	9A		TXS	
0343	A9	02	LDA ≠ 02	Forzar STACK para que contenga la dirección 0205.
0345	48		PHA	
0346	A9	05	LDA ≠ 05	
0348	48		PHA	
0349	4C	D002	JMP 02D0	A BAJADA.

CONCLUSIONES

Los proyectos propuestos en este capítulo han sido diseñados para ser probados en microcomputadores básicos y económicos, como es el caso del KIM-1 ó el de la casa Heathkit. Se conseguirán resultados más rápidos y se podrá acometer a diseños más complejos, si se dispone de microcomputadores más potentes. Así, si se emplea el AIM-65 basado también en el microprocesador R 6502, cuya vista se ofrece en la figura 8-82, y que se describe su arquitectura en el otro libro de esta obra, se podrán cargar los programas en lenguaje Ensamblador e incluso en Basic, se obtendrá una copia escrita de todas las operaciones mediante una impresora térmi-

Fig. 8-82.–Fotografía del microcomputador AIM-65.

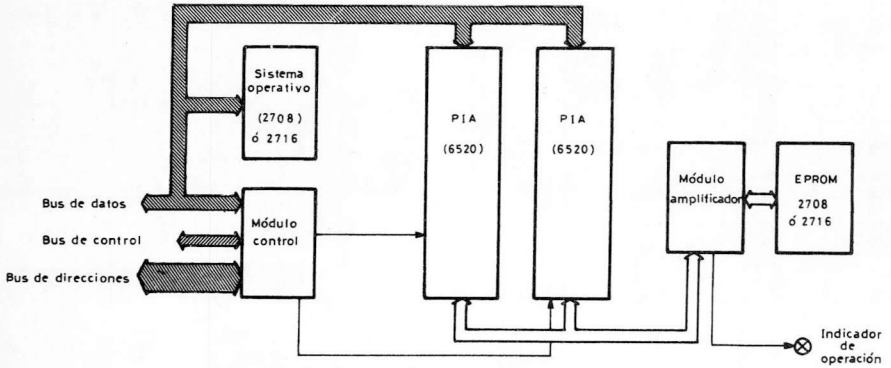


Fig. 8-83.–Diagrama por bloques de la tarjeta destinada a la grabación de EPROM en combinación con el AIM-65.

ca, se podrán manejar 2 magnetofones y se dispondrá de numerosas entradas y salidas entre las características más relevantes.

Una vez probados y depurados los programas y el hardware de los proyectos, es necesario grabar los programas de trabajo para implementar los prototipos sobre memorias no volátiles, del tipo PROM, para lo cual se precisa de un "programador de PROM". Muchos microcomputadores, co-

mo el AIM-65, disponen de una tarjeta opcional que permite grabar a las memorias PROM. Para el microcomputador AIM-65, COMELTA fabrica la tarjeta CR-108, que faculta la grabación de las memorias EPROM 2708 y 2716. Dicha tarjeta dispone de un conector DIN, compatible con el AIM-65, y de un zócalo especial de fuerza de inserción nula, para la EPROM a grabar. El sistema operativo reside sobre otra EPROM en la misma tarjeta. La figura 8-83 muestra el diagrama por bloques de dicha tarjeta.

La figura 8-84 muestra dos fotos de la tarjeta CR-108, con la que se pueden realizar las siguientes operaciones:

1) Lectura de EPROM original. Verificación y chequeo con listado.

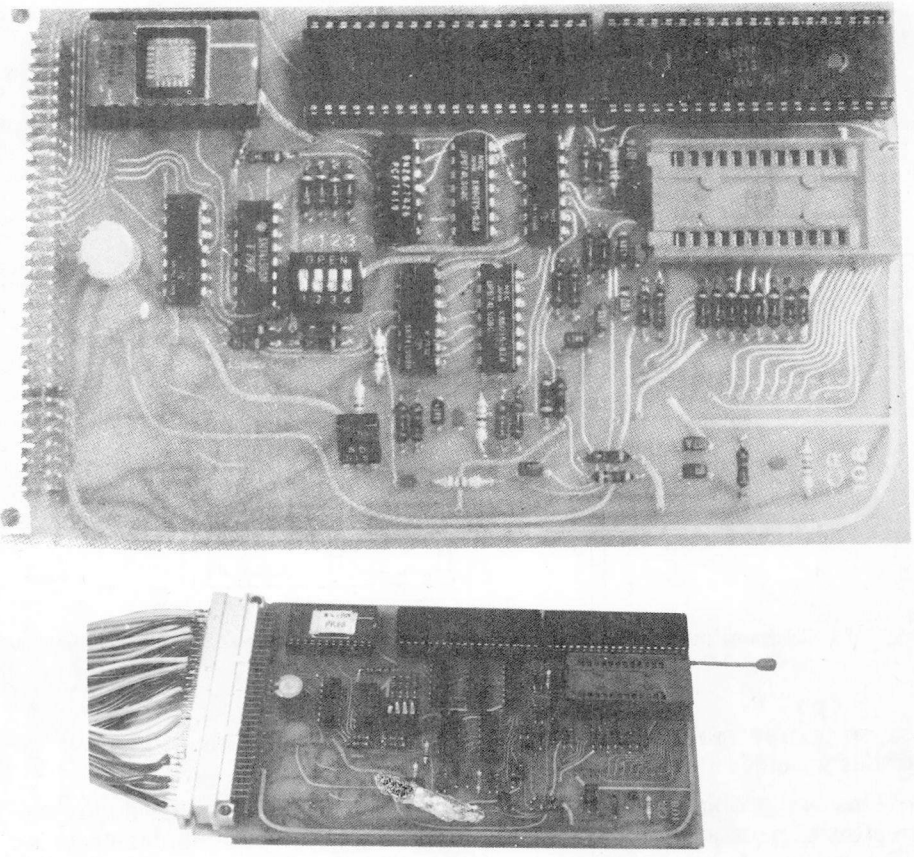


Fig. 8-84.—Tarjeta de programación de EPROM (foto superior) y conector que se inserta para su adaptación al AIM-65. (Foto interior.)

2) Programación y reprogramación.

3) Indicación del número de errores en cada operación.

4) Realización de operaciones en modo conversacional con el operador.

Superando al AIM-65 (capacidad de memoria, potencia, lenguajes de alto nivel, gran variedad de periféricos, etc.) caben destacar entre los microcomputadores de "carácter personal", que emplean como CPU al microprocesador R 6502, al PET de Commodore, al "CHALENGER" de Ohio Scientific y al APPLE, con monitores de vídeo incorporados, unidad de cinta magnética y capacidad para adaptar múltiples periféricos, como impresoras, floppy-disk, etc.

APENDICE

EJEMPLOS DE PROGRAMACION CON EL MICROPROCESADOR 2650

INTRODUCCION

Con objeto de ofrecer alguna aplicación sobre la programación del microprocesador de Signetic, se escriben en este artículo algunas formas de escribir rutinas de tiempos de retardo por programa. Ejemplos de aplicación para los microprocesadores INTEL se encontrarán en el otro libro que compone esta obra sobre microprocesadores.

RUTINAS GENERALES DE RETARDO

En las aplicaciones de los microprocesadores los tiempos de retardo se precisan muy a menudo. Un ejemplo típico lo constituye el tiempo de retardo que necesita un acoplamiento serie para un teletipo. Aunque pueden conseguirse los tiempos de retardo mediante contadores, monoestables, multivibradores, etc., es más simple y económico utilizar una pequeña rutina por programa.

En el microprocesador 2650 los registros internos se pueden utilizar como contadores. En general, se puede realizar un retardo, colocando un número N en un contador y decrementándolo uno a uno hasta que sea cero. Si el decremento del número ocupa un período de reloj, entonces el tiempo total de retardo son N períodos de reloj.

En el caso de utilizar los registros del microprocesador como contadores, las instrucciones más útiles para el decremento son las de Branch sobre la decrementación de Registro (BDRR y BDRA), según que el direccionamiento sea el relativo o el absoluto, respectivamente, que también comprueban si el contenido de un registro es cero.

En la figura 8-85 se recoge el ordinograma general para obtener una rutina de retardo, según método propuesto por la Revista Miniwatt. Esta rutina consiste en una zona establecida y un bucle o lazo de contaje. El lazo contador se ejecuta n veces y el programa previo establecido sólo una vez, consiguiéndose un tiempo de retardo:

$$t_r = t_e + n \cdot t_{lc}$$

De esta forma es posible incrementar el tiempo de retardo, bien sea incrementando n o haciendo t_{lc} más largo. Esto último se puede hacer in-

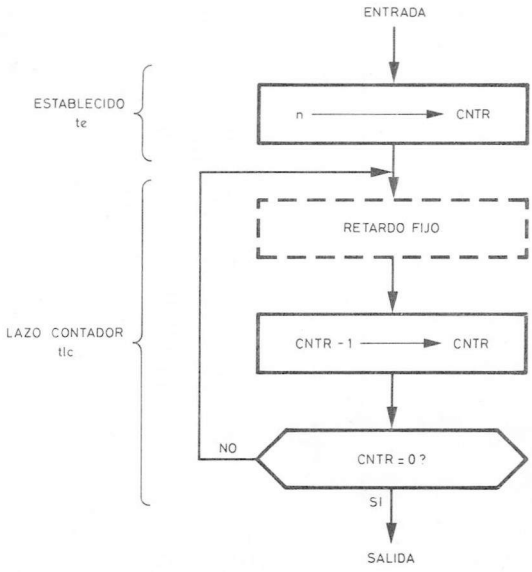


Fig. 8-85.– Ordinograma general que establece la rutina de temporización.

sertando un retardo fijo tal como la instrucción No Operation (NOP) en el lazo contador.

El programa de la subrutina mostrada en el diagrama de la figura 1 es el siguiente:

	LODI, Rx n	Cargar n dentro del registro Rx;	6 pr*
LAZO	NOP	No Operación; retardo fijo de 6 períodos de reloj (pr)	6 pr
	BDRR, Rx LAZO	Decrementar Rx; bifurcar a lazo si el resultado no es cero.	9 pr

*pr = períodos de reloj.

Con un NOP, el tiempo de retardo es: $t_r = (6 + 15 \cdot n)$ pr. Sin el NOP, el tiempo de retardo es: $t_t = (6 + 9 \cdot n)$ pr. El máximo tiempo de retardo se obtiene cuando Rx es cargado con cero, ya que Rx hará el ciclo de los 256 estados posibles. Cuando Rx = R0, la instrucción LODI, R0 0 se puede sustituir por la instrucción EORZ R0 que ahorrará un octeto de código.

Otra forma de efectuar el tiempo de retardo es repetir varias veces el lazo contador de la figura 8-85. Esto se puede hacer por repetición de las instrucciones o contando en otro registro las veces que se efectúa el lazo.

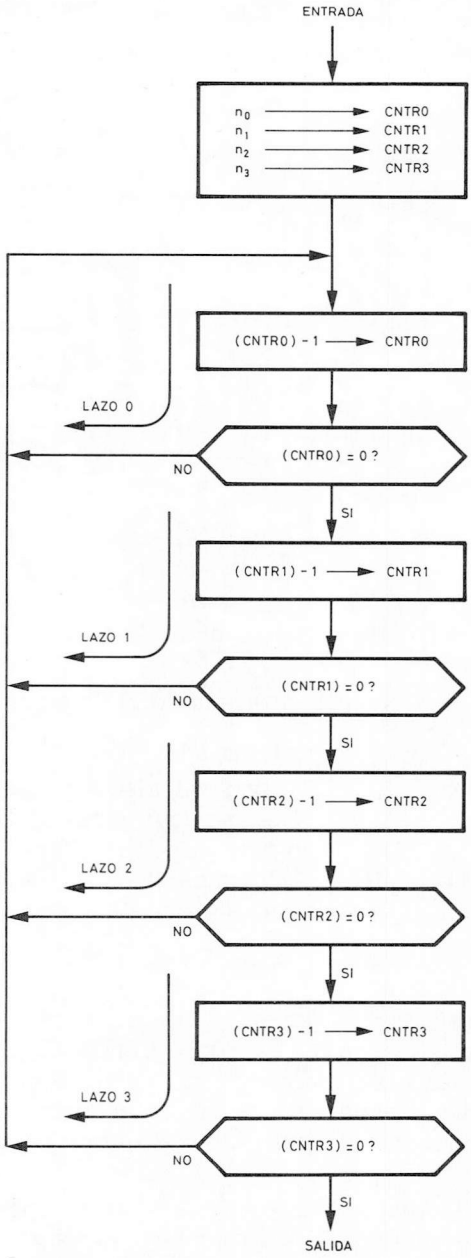


Fig. 8-86.–Ordinograma de una rutina de retardo.

Este último método se puede ampliar hasta ocupar cuatro registros internos. La figura 8-86 muestra el diagrama de flujo de una rutina de retardo empleando esta técnica.

El número de veces que el microprocesador ejecuta los diferentes lazos, que se muestran en la figura 8-80 son:

lazo 3: n_3

lazo 2: $n_2 + (n_3 - 1)\,256$

lazo 1: $n_1 + (n_2 - 1)\,256 + (n_3 - 1)\,256^2$

lazo 0: $n_0 + (n_1 - 1)\,256 + (n_2 - 1)\,256^2 + (n_3 - 1)\,256^3$

Así pues, el tiempo de retardo de esta subrutina es:

$$t_r = \{24 + [n_0 + n_1 + (n_1 - 1)\,256 + n_2 + (n_2 - 1)(256 + 256^2) + n_3 + (n_3 - 1)(256 + 256^2 + 256^3)]\,9\}\ pr$$

Si Rx se carga con un cero, entonces n = 256 en la fórmula.

Rutina	*Tiempo de retardo*
a	$t_r = (6 + 9 \cdot n_0)\ pr$
b	$t_r = (6 + 15 \cdot n_0)\ pr$
c	$t_r = (2.310 + 9 \cdot n_0)\ pr$
d	$t_r = \{12 + [n_0 + n_1 + (n_1 - 1)\,256]\,9\}\ pr$
e	$t_r = \{18 + [n_0 + n_1 + (n_1 - 1)\,256 + n_2 + (n_2 - 1)(256^2 + 256)]\,9\}\ pr$
f	$t_r = \{24 + [n_0 + n_1 + (n_1 - 1)\,256 + n_2 + (n_2 - 1)(256^2 + 256) + n_3 + (n_3 - 1)(256^3 + 256^2 + 256)]\,9\}\ pr$

TABLA 1

Rutina	Tiempo de retardo (pr)		Pasos de retardo (pr)	Número de octetos	Número de registros	Programa		
	mín*	máx						
a	15	2310	9	4	1		LODI, R0 n_0	
						LAZO	BDRR, R0 n_0	LAZO
b	21	3846	15	5	1		LODI, R0 n_0	
						LAZO	NOP	
							BDRR, R0	LAZO
c	2319	4614	9	6	1		LODI, R0 n_0	
						LAZ 1	BDRR, R0	LAZ 1
						LAZ 2	BDRR, R0	LAZ 2
d	30	592.140	9	8	2		LODI, R0 n_0	
							LODI, R1 n_1	
						LAZO	BDRR, R0	LAZO
							BDRR, R1	LAZO
e	45	$\approx 151{,}6 \times 10^6$ **	9	12	3		LODI, R0 n_0	
							LODI, R1 n_1	
							LODI, R2 n_2	
						LAZO	BDRR, R0	LAZO
							BDRR, R1	LAZO
							BDRR, R2	LAZO
f	60	$\approx 38{,}8 \times 10^9$ ***	9	16	4		LODI, R0 n_0	
							LODI, R1 n_1	
							LODI, R2 n_2	
							LODI, R3 n_3	
						LAZO	BDRR, R0	LAZO
							BDRR, R1	LAZO
							BDRR, R2	LAZO
							BDRR, R3	LAZO

* pr = período de reloj. Para 1 MHz, 1 pr = 1 μs
** Para 1 MHz éste es de unos 2,5 minutos.
*** Para 1 MHz, éste es de unas 10,46 horas.

DICCIONARIO DE LA TERMINOLOGIA INGLESA RELACIONADA CON MICROCOMPUTADORES

A

ABORT: "Abortar".
Hace referencia a la función que sirve para detener el programa en ejecución y devuelve el control al operador o al sistema operativo. En muchos microcomputadores dicha operación se lleva a cabo mediante el comando RESET.

ACCES TIME: "Tiempo de acceso."
Es el tiempo requerido para la recepción de un dato válido, desde un elemento de memoria, después de producirse una señal de lectura. Otra forma de expresar este concepto es la siguiente: "Intervalo de tiempo existente entre la petición de información y el instante en que esta información está disponible para ser usada."

ACCUMULATOR: "Acumulador."
Se trata de un registro (en ocasiones varios) asociado con la Unidad Lógico Aritmética (ALU), que temporalmente almacena sumas, expresiones aritméticas y resultados lógicos procedentes de la ALU.

ACCURACY: "Factor de precisión".
Es el grado de libertad existente frente al error. Generalmente expresa el porcentaje de error máximo. Por ejemplo, una magnitud de 6 dígitos es menos precisa que otra de 8, aunque el valor expresado con 6 dígitos, correctamente tratado, pueda ser más exacto que el de 8 dígitos.

ACK: "Acuse de recibo." (Acknowledge.)
Mensaje de reconocimiento, enviado a través de un enlace de comunicación, para indicar que la recepción de los datos ha sido correcta.

A/D: "Analógico/Digital."
Se refiere a la conversión de señales analógicas en digitales. Ver "Analog to digital converter".

ADC: "Conversor analógico/digital."
Se trata de un circuito encargado de transformar las señales analógicas en digitales.
También puede presentar el nemónico de una instrucción de suma con acarreo, usada en los microprocesadores.

ADDER: "Sumador."
Elemento aritmético que proporciona en su salida la suma de dos o más números, presentes en sus entradas, así como el acarreo.

ADDRESS: "Dirección."
Valor empleado por la CPU para indicar la posición de una palabra en la memoria. Es típico en los microprocesadores que las direcciones dispongan de 16 bits de longitud, lo cual permite posicionar hasta 64 K posiciones (1 K = 1.024).

ADDRESS BUS: "Bus de direcciones."
Juego de líneas (generalmente 16), usado para transmitir direcciones, normalmente desde el microprocesador a la memoria o los elementos de Entradas/Salidas.

ADDRESS DECODING: "Decodificación de direcciones".
Proceso de selección de una dirección específica o de un campo de direcciones, que habilita a un único elemento.

ADDRESS FORMAT: "Formato de dirección."
Orden de las partes de la dirección de una instrucción, tales como las necesarias para identificar el canal, módulo, pista, etc., en un sistema de disco.

ADDRESS REGISTER: "Registro de dirección."
Registro en el que se almacena una dirección.

ADDRESSING MODES: "Modos de direccionamiento."
Son los diversos métodos que existe para especificar una dirección como parte de una instrucción. Entre los modos de direccionamiento más conocidos destacan el "directo", "indirecto", "inmediato" e "indexado".

ALGORITH: "Algoritmo."
Procedimiento para resolver un problema paso a paso. En principio, se propone el problema y, a continuación, se proyecta un algoritmo para su solución.

ALPHANUMERIC: "Alfanumérico."
Juego de caracteres alfabéticos y numéricos.

ALU: "Unidad Lógica Aritmética."
Ver más adelante su explicación.

ANALOG: "Analógico."
Magnitud variable de forma continua. Concepto opuesto a digital.

ANALOG REPRESENTATION: "Representación analógica."
Representación que está continuamente variando o es variable en cualquier momento.

ANALOG TO DIGITAL CONVERTER: "Conversor analógico a digital."
Elemento que convierte voltajes y corrientes analógicas en formas digitales, empleadas por los sistemas computadores. Permite al computador el procesamiento de señales reales exteriores.

ARCHITECTURE: "Arquitectura."
Estructura lógica de un sistema computador.

ARITHMETIC AND LOGIC UNIT (ALU): "Unidad Lógico Aritmética."
Se trata de uno de los tres componentes esenciales del microprocesador. Los otros dos son el bloque de control lógico y el juego de registros. La ALU permite realizar diferentes tipos de sumas, restas y operaciones lógicas, como por ejemplo, la operación AND entre los contenidos de dos registros o el borrado del contenido de un registro.

ARITHMETIC SHIFT: "Desplazamiento aritmético."
Equivalente a la multiplicación de un número por la potencia entera (positiva o negativa) de la base.

ARRAY LOGIC: "Conjunto lógico."
Red lógica, cuya configuración es un rectángulo de intersecciones de sus líneas de entrada y de salida. Los elementos están conectados a alguna de dichas intersecciones y la red funciona normalmente como un codificador o un decodificador.

ASCII (American Standard Code for Information Interchange): "ASCII (Código americano estándar para el intercambio de información."
Consiste en un código de caracteres utilizados para la representación de la información en muchos sistemas computadores.
Este código usa un conjunto de 7 bits para codificar un carácter (8 bits incluyendo el de paridad). Es sinónimo de USASCII.

ASSEMBLE: "Ensamblar."
Ensamblar es preparar un programa en lenguaje máquina a partir del programa en lenguaje simbólico. Esto se logra sustituyendo los códigos de operación y direcciones simbólicas por los códigos y direcciones absolutos.

ASSEMBLER PROGRAM: "Programa ensamblador."
Es un programa del computador que convierte un programa en lenguaje Ensamblador en un programa objeto ejecutable (codificado en binario). Traduce el lenguaje Ensamblador (nemónico) al lenguaje máquina (binario).

ASSEMBLY LANGUAGE: "Lenguaje ensamblador."
Lenguaje orientado a la máquina (de bajo nivel). Normalmente, un programa es escrito como una serie de pasos, empleando símbolos nemónicos, que sugieren el carácter de la instrucción. Este lenguaje es traducido al lenguaje máquina mediante el programa Ensamblador. Los programas en el lenguaje ensamblador son leídos por el Ensamblador y convertidos al lenguaje de máquina, ejecutable durante el período de ensamblaje. El lenguaje ensamblador es más fácil de recordar y manipular que el lenguaje máquina.

ASYNCHRONOUS: "Asíncrono."
Se refiere a cualquier circuito o sistema que no está sincronizado por una señal común de reloj.

ASYNCHRONOUS DEVICE: "Elemento asíncrono."
Dispositivo cuya velocidad de operación no está ligada a la frecuencia de alguno de los elementos del sistema al que está conectado.

B

BACKPLANE: "Tarjeta base o tarjeta principal." (A veces se llama *motherboard*.)
Es la tarjeta de circuito impreso, sobre la que se insertan otras tarjetas que forman el sistema. Generalmente contiene los buses del sistema.

BASE: "Base."
Ver Radix más adelante.

BASIC: "Basic."
Se trata de un lenguaje muy sencillo de aprender y de usar, que se utiliza en la mayor parte de los sistemas microcomputadores.

BAUD RATE: "Medidas de baud."
Es una medida del flujo o velocidad de datos, indica el número de elementos de señal por segundo. Cuando cada elemento transporta un bit, la medida de baud equivale al número de bits por segundo (bps). Por ejemplo, los teletipos transmiten a 110 baud. Cada carácter es de 11 bits y el teletipo transmite 10 caracteres por segundo.

BAUDOT CODE: "Código BAUDOT."
Código de información usado en la transmisión de datos.

BCD (BINARY CODED DECIMAL): "BCD (Decimal codificado en binario)."
Es una representación, mediante 4 bits, de los dígitos decimales de 0 al 9. Seis de las 16 posibles combinaciones no se emplean. Por lo general 2 dígitos BCD forman un byte.

BENCHMARK: "Punto de referencia."
Método usado para la medida del funcionamiento del computador en una situación bien definida.

BENCHMARK PROBLEM: "Problema de referencia."
Problema utilizado para evaluar la ejecución de software, el hardware, o bien ambos.

BIDIRECTIONAL: "Bidireccional."
Indica que el flujo de la señal puede producirse en ambos sentidos. Es común que los buses bidireccionales sean triestados o TTL de colector abierto.

BYNARY: "Binario."
Es un sistema de numeración, que emplea como base el 2, en contraste con el decimal, que emplea como base el 10. El sistema binario precisa solamente de dos símbolos, el 0 y el 1. Dos es expresado en binario por 10.

BINARY SEARCH: "Exploración binaria."
Técnica en la que se divide por 2 cada intervalo explorado en cada paso.

BIT: "Bit."
Contracción de binary digit (dígito binario). Consiste en un simple dígito de un número binario. Un número binario es 0 o 1. Un bit es la menor unidad de información en un sistema binario de anotación.

BIT-SLICE: "Rodaja de bit."
Método que implementa una CPU de n "bit-slice". Normalmente n = 4. Un procesador de 32 bits podrá fabricarse con 8 CPU de 4 bit-slice.

BLOCK: "Bloque."
1.—Conjunto de cosas, tales como palabras, caracteres o dígitos, manejados como una unidad.
2.—Colección de registros contiguos, utilizados como una unidad.
3.—Grupo de bits o dígitos binarios, transmitidos como una unidad.
4.—Grupo de caracteres contiguos, empleados como una unidad.

BLOCK DIAGRAM: "Diagrama por bloques."
Diagrama de un sistema, instrumento u ordenador, en el cual las partes principales se representan mediante figuras geométricas asociadas de forma conveniente, para mostrar las funciones básicas y las relaciones funcionales entre las partes.

BOARD TESTER: "Probador de tarjetas."
Producto programado para estimular o alimentar los circuitos sobre tarjetas y probar sus respuestas. Sirve para detectar averías y simplificar la reparación de circuitos montados sobre tarjetas.

BOOLEAN LOGIC: "Lógica de Boole."
Nombre derivado de George Boole, quien definió las operaciones aritméticas binarias y las lógicas, tales como AND, OR, NOR y OR-Exclusivo.

BOOTSTRAP: "Comando de entrada."
Programa usado para iniciar el computador. Usualmente borra la memoria, activa los elementos de I/O y carga el sistema operativo.

BORROW: "Acarreo negativo."
Carry negativo aritméticamente.

BOUNCE: "Rebote."
Oscilaciones y ruidos generados cuando un interruptor mecánico es abierto y cerrado. Ver Debounce.

BRANCH: "Bifurcación."
Ver Jump.

BRANCHING: "Bifurcando."
Método de selección en base a los resultados de la siguiente operación a ejecutar, mientras el programa está ejecutándose.

BREAKPOINT: "Punto de parada."
Elemento de hardware o software que detiene el programa y salva el estado actual de los registros de la máquina bajo condiciones determinadas por el usuario.

BUBBLE MEMORY: "Memoria de burbuja."
Memoria que emplea material magnético en el que una zona microscópica de magnetización, llamada burbuja, almacena un bit sobre un material tal como el granate magnético. Son muy importantes, porque son memorias de estado sólido no volátiles de gran capacidad de almacenamiento.

BUFFER: "Circuito tampón, amplificador separador."
En un circuito integrado, que restaura o conduce al nivel lógico que recibe. Pueden ser triestados, en cuyo caso su salida está "flotante" hasta que no se le aplica la señal Enable. También es un dispositivo de almacenamiento, usado para compensar la diferencia de velocidad en la transmisión de información o en el tiempo en que ocurren los sucesos, cuando se transmite información desde un dispositivo a otro.

BUG: "Error."
Se denomina a un error. De aquí que "debugging" signifique la técnica de depuración o eliminación de errores.

BUS: "Bus."
Se trata de un colector de señales que tienen la misma función. La mayoría de los microprocesadores usa tres buses: el de datos, el de direccionamiento de la memoria y el de control. Se trata de un grupo de conductores eléctricos, que proporcionan una comunicación entre 2 o más dispositivos, tales como entre un procesador central, memoria o periféricos.

BUS CONFLICT: "Bloqueo o conflicto en el bus."
Problema que surge cuando la salida de dos elementos de estados lógicos opuestos son colocadas al mismo tiempo sobre un bus triestado.

BUS CONTROLLER: "Controlador del bus."
Genera señales de control y gobierno del bus.

BUS DRIVER: "Conductor del bus."
Es un circuito integrado, que se añade al bus para procurar la suficiente adaptación entre la CPU y los demás elementos conectados al bus. Son necesarios porque la carga capacitiva retarda la velocidad de los datos e impide la correcta secuencia del microprocesador.

BUS TERMINATION: "Terminación de bus."
Método para la prevención de reflexiones en el final de un bus. Sólo es necesario en sistemas de alta velocidad.

BYTE: "Byte."
Grupo de 8 bits. Puede usarse para representar un carácter. Las instrucciones de los microcomputadores necesitan uno, dos o tres bytes. Una palabra consta de uno o más bytes.

C

CALCULATOR: "Calculadora."
Procesador de datos, especialmente útil para realizar operaciones aritméticas que requieren frecuentes intervenciones del operador humano.

CALL: "Llamada."
Salto a una subrutina. Se ejecuta un salto a una dirección determinada, pero se salva el contenido del Contador de Programa (usualmente en el stack) para retornar el Programa Principal cuando finaliza la subrutina.

CALLING: "Llamando."
Ver "Subroutine CALL".

CARRY FLAG: "Señalador del acarreo."
Consiste en un bit indicador del registro de estado del microprocesador, que indica el acarreo o sobrepasamiento de capacidad de una operación efectuada por la Unidad Lógico-Aritmética.

CARRY-LOOK-AHEAD: "Sumador especial."
Tipo de sumador en el cual las entradas a varios estados son examinados y los acarreos correspondientes son generados simultáneamente.

CASCADE CONECTION: "Conexión en cascada."
Dos o más dispositivos similares, colocados uno tras otro, con la salida de uno conectada a la entrada del siguiente.

CCD (CHARGE COUPLED DEVICE): "CCD (Elemento de acoplo de carga)."
Tecnología de almacenamiento en serie, que emplea condensadores MOS. Consiste en pequeñas placas de aluminio sobre un substrato de silicio.

CHANNEL: "Canal."
Camino a través del cual pueden ser enviadas señales; por ejemplo, canal de datos, canal de salida.

CHARACTER: "Carácter."
Letra, dígito u otro símbolo que se usa como parte de la organización, control o representación de un dato. Un carácter se presenta a menudo bajo la forma de una ordenación especial de rasgos adyacentes o conectados.

CHARACTER GENERATOR: "Generador de caracteres."
Circuito que conforma letras o números sobre un display o una impresora.

CHECK BIT: "Bit de prueba."
Dígito binario de control; por ejemplo, bit de paridad.

CHEKER BOARD: "Tarjeta de prueba."
Se trata de un modelo de 1 y 0, que sirve para probar la memoria al almacenarlo en el conjunto de células del conjunto de la memoria.

CHEKSUM: "Verificación."
Método usado para la verificación de la integridad de los datos cargados en el computador.

CHIP: "Chip" o "pastilla".
Nombre con el que se designa comúnmente a todos los circuitos integrados.

CHIP ENABLE: "Permiso al chip."
Ver más adelante Chip Select.

CHIP SELECT: "Selección de chip."
Usualmente permite la actuación de las salidas de un chip driver triestado. La mayoría de los chips LSI disponen de una o más líneas para la selección del chip. La línea CS (chip select) se usa para seleccionar un chip entre varios.

CENTRAL PROCESSING UNIT: "Unidad Central de Proceso."
Módulo del computador que realiza la búsqueda, decodificación y ejecución de instrucciones. Consta de una unidad de control, una ALU y otros elementos, tales como registros, clocks, etc. Recibe la denominación de CPU. Parte del sistema de un ordenador que contiene el almacenamiento principal, la unidad aritmética y grupos de registros especiales. Realiza operaciones aritméticas y facilita señales de tiempo y otras operaciones de manejo interno.

CIRCUIT: "Circuito."
En comunicaciones, medio a través del cual se puede realizar la comunicación entre dos puntos, incluyendo canales de ida y vuelta.

CLEAR: "Borrado."
Coloca a un circuito en un estado conocido, generalmente en cero.

CLEARING: "Borrado."
Puesta a ceros binarios, tanto para un registro como para un bit individual.

CLOCK: "Reloj."
Fuente de referencia de tiempo en un sistema. Un clock proporciona los impulsos, que sincronizan y disparan regularmente la secuencia de acontecimientos.

CLOSED-LOOP: "Bucle cerrado."
Circuito que opera con realimentación, cuyas entradas son función de sus salidas.

CODE: "Código."
Tal como el código del lenguaje máquina, o el proceso de conversión desde un lenguaje a otro. Conjunto de reglas que especifican la forma en que los datos deben ser representados. En telecomunicaciones, sistema de reglas y conversiones de acuerdo con las cuales se pueden formar, transmitir, recibir y procesar las señales que representan los datos.

COMBINATIONAL LOGIC SISTEM: "Lógica combinacional."
Circuito cuyas salidas quedan determinadas exclusivamente por el estado de las entradas. También se llama lógica combinatoria.

COMMENT FIELD: "Campo del comentario."
Campo de una instrucción destinado a los comentarios. Los comentarios son ignorados por el compilador o el ensamblador, cuando se convierte a código máquina el programa.

COMMUNICATION CONTROL CHARACTER: "Carácter control de comunicación."
Carácter de control, ideado para controlar o facilitar la transmisión de datos a través de las redes de comunicación.

COMPILE: "Compilar."
Preparar un programa en lenguaje máquina a partir de un programa de ordenador escrito en otro lenguaje de programación. Esta función se realiza haciendo uso de la estructura lógica del programa o generando más de una instrucción máquina para cada frase simbólica.

COMPILER: "Compilador."
Programa de traducción que convierte instrucciones de alto nivel en un conjunto de instrucciones binarias (Código máquina) para su ejecución. Cada lenguaje de alto nivel requiere un compilador o un interpretador. Un compilador traduce un programa completo, que luego debe ejecutar.

COMPLEMENT: "Complementar."
Proceso de cambiar los 1 a 0 y los 0 a 1. Complemento a 1.

COMPLEMENT NOTATION: "Notación por complementos."
Sistema de notación donde los números binarios positivos son idénticos a los números positivos en signo y magnitud, pero los números negativos son el complemento exacto de la magnitud del correspondiente valor positivo.

COMPUTER: "Computador u ordenador."
Sistema de proceso de datos de aplicación general, que incorpora una CPU, memoria I/O y fuente de alimentación.

CONDITIONAL BRANCH INSTRUCCION (Also CONDITIONAL JUMP). "Instrucción de bifurcación condicional (también salto condicional)." Instrucción condicional de BRANCH (Bifurcación) (igualmente, "salto condicional"). Esta instrucción da lugar a transferir a otra instrucción de un programa, que no es la próxima instrucción en secuencia, "solamente" si, examinada una condición específica, es satisfecha. Si la condición no es satisfecha, es ejecutada la instrucción siguiente en la secuencia del programa.

CONDITION CODE: "Código de condición."
Se refiere a un grupo limitado de condiciones del programa, como el acarreo, el sobrepasamiento, el signo, etc., que son necesarias comprobar para la ejecución de determinadas instrucciones. Los códigos están contenidos en un registro de código de condición, conocido también como registro de flags. Un tipo de instrucciones, en los que intervienen los códigos de condición, son los "branch condicionales".

CONDITIONAL JUMP OR CALL: "Salto o llamada condicional."
Se trata de un tipo de instrucción que, cuando aparece en el programa, puede producir el paso a la siguiente instrucción o el salto a otra instrucción cualquiera, dependiendo de una determinada condición. Salto o bifurcación que ocurre, si una condición especificada se cumple.

CONDITIONAL LOOP: "Bucle condicional."
Un loop (bucle) puede o no ser ejecutado, dependiendo de una condición examinada por la instrucción condicional de branch (bifurcación).

CONSTANT: "Constante."
Un valor fijo.

CONTENTS: "Contenidos."
Valor binario almacenado en un flip-flop, registro o localización de memoria.

CONTROL BLOCK: "Bloque de control."
Circuitos que realizan funciones de control de la CPU. Son responsables de la decodificación de instrucciones y la generación de señales internas de control, que efectúan las operaciones solicitadas.

CONTROL BUS: "Bus de control."
Conjunto de las líneas de control en un sistema computador. Proporcionan la sincronización y la información necesaria para realizar y ejecutar el programa.

CONTROL CHARACTER: "Carácter de control."
Carácter cuya inclusión en un contexto particular inicia, modifica o para una operación de control; por ejemplo, un carácter que controla el acarreo, un carácter que controla la transmisión de datos en una red de comunicaciones...

CONTROL PROGRAM: "Programa de control."
Secuencia de instrucciones que llevan a la CPU a través de diversas operaciones. Este programa está almacenado de forma permanente en ROM, a la que puede acceder la CPU durante una operación. Generalmente, esta ROM está localizada en el mismo chip del microprocesador. Se llama también microprograma o microcódigo.

CONTROLLER: "Regulador."
Subsistema digital responsable de implementar "cómo" va a funcionar un sistema. No debe ser confundido con "Timing" pues timing dice al sistema "cuándo" debe efectuar su función.

CORE: "Núcleo."
Pequeño toro de ferrita, magnético, encargado de almacenar un bit de información. No es volátil.

CPS: "cps."
Caracteres por segundo.

CPU: "CPU o UCP".
Unidad Central de Proceso. Normalmente contiene el TIMING y sección de control, además de las secciones aritmética y lógica de un computador.

CRASH: "Quiebra o rotura."
Fallo en el hardware o software que origina el paro del sistema o la pérdida en un loop.

CRC: "CRC."
(Ciclic Redundancy Check. Prueba de redundancia cíclica.) Carácter de control del ciclo de redundancia.

CRITICAL RACE: "Carrera crítica."
Situación de tiempo relacionada con una operación asíncrona. Una "carrera" puede ocurrir cuando dos variables son solicitadas para cambiar estados simultáneamente. "Crítico", se refiere al resultado que determinará el estado de la máquina.

CROSS-ASSEMBLER: "Ensamblador cruzado."
Ensamblador que "corre" sobre un procesador, cuyo lenguaje ensamblado es diferente al lenguaje en que ha sido ensamblado el programa.

CROSSTALK: "Diafonía o efecto de acoplamiento perjudicial."
Interferencia entre 2 señales. Interferencia que aparece en un determinado canal, pero que ha tenido su origen en otro canal.

CRT TERMINAL: "Terminal CRT."
Terminal de computador que usa como visualizador un tubo de rayos catódicos CRT y un teclado, conectado generalmente en serie con el computador.

CURRENT TRACER: "Marcador de corriente."
Instrumento de reparación usado para determinar el flujo de corriente en circuitos lóficos.

CYCLE TIME: "Tiempo de ciclo."
Tiempo total requerido por un elemento de memoria para completar un ciclo de lectura o escritura y quedar disponible de nuevo.

CYCLE REDUNDANCY CHECK: "CRC, prueba de reduncancia cíclica."
Polinomio binario usado para generar información de prueba sobre los bloques de datos.

D

D/A: "D/A".
Ver convertidor digital a analógico.

DAC: "DAC."
Conversor digital analógico.

DATA: "Dato".
Término general que denota cualquiera de todos los hechos, números, letras y símbolos, que describen un objeto, idea, condición, situación u otros factores. Son los elementos básicos de información que pueden ser procesados o producidos por un ordenador. A veces se considera a los datos únicamente como expresiones numéricas, pero pueden tener otras formas.

DATA ACQUISITION: "Adquisición de datos."
Colección de datos procedentes de sensores externos, generalmente en forma analógica.

DATA BASE: "Base de datos."
Organización sistemática de líneas de datos, para facilitar su acceso y recobrarlos.

DATA BUS: "Bus de datos."
Juego de líneas por los que se transportan los datos. Generalmente es bidireccional y triestado.

DATA DOMAIN: "Campo de datos."
Análisis o visualización de señales en las que sólo se considera un valor digital y no los correspondientes al voltaje y al tiempo.

DATA PROCESSING: "Proceso de datos."
Ejecución de una secuencia sistemática de operaciones realizadas con los datos. Sinónimo de "proceso de información".

DATA PROCESSOR: "Procesador de datos."
Dispositivo capaz de realizar el proceso de información, incluyendo calculadoras de pupitre, máquinas de tarjetas perforadas y ordenadores. Sinónimo de procesador.

DEBOUNCING: "Eliminación de rebotes."
Eliminación de las señales de rebote, características de los interruptores mecánicos. La supresión de rebotes puede llevarse a cabo por hardware o por software.

DEBUG: "Depurar."
Detectar, localizar y corregir errores en una rutina o funcionamiento incorrecto en un ordenador. Sinónimo de "TROUBLESHOOT".

DEBUGGER: "Depurador."
Programa diseñado para facilitar la depuración del software. En general, proporciona breakpoints (puntos de parada), dump (transferencia de zonas de memoria a otras) y facilidades para examinar y modificar registros y posiciones de memoria.

DEBUGGING: "Depuración."
Procedimiento para eliminar los errores de hardware y de software en un sistema.

DECIMAL: "Decimal."
Perteneciente a una característica o propiedad que involucra una selección, elección, o condición en la que hay 10 posibilidades. Al sistema de numeración de base 10.

DECIMAL DIGIT: "Dígito decimal."
En la notación decimal es uno de los caracteres de 0 a 9.

DECISION BLOCK: "Bloque de decisión."
Elemento o figura en un diagrama de flujo, que indica una o más posibles acciones basadas en el resultado de una decisión. Cuando se traslada a lenguaje ensamblador, incluirá una instrucción Condicional de Branch (Bifurcación).

DECODER: "Decodificador."
Elemento lógico que decodifica sus entradas binarias. Un decodificador de 3 entradas dispondrá de $2^3 = 8$ posibles salidas.

DECREMENT: "Decrementar."
Instrucción de programación que decrementa el contenido de una posición de almacenamiento, o de un registro.

DEDICATED: "Dedicado."
Un microprocesador "dedicado" consiste en un tipo que ha sido especialmente programado para realizar una

aplicación determinada, como la medida del peso u otra cualquiera. Las ROM por naturaleza son memorias dedicadas.

DEVELOPMENTS SYSTEM: "Sistema de desarrollo."
Sistema microcomputador con todas las facilidades o ayudas necesarias para el desarrollo del hardware y software de un determinado microprocesador. Generalmente consta de un sistema microcomputador con pantalla CRT, impresora, masa de memoria (normalmente dos floppy-disk), programador de PROM y circuito emulador.

DEVICE CONTROL CHARACTER: "Carácter de control de dispositivo."
Carácter de control diseñado para el control de los dispositivos auxiliares asociados con un proceso de datos o un sistema de telecomunicaciones, normalmente empleado para conectar o desconectar los dispositivos auxiliares (terminales).

DIAGNOSTIC: "Diagnóstico."
Proceso de detección y aislamiento de un mal funcionamiento o de un error.

DIGIT: "Dígito."
Signo o símbolo usado para especificar una cantidad de información por sí mismo o en combinación con otros números de su juego; 2, 3, 4 y 5 son dígitos. La base debe ser especificada y cada valor de dígito asignado.

DIGITAL: "Digital."
Los elementos que tienen estados discretos. La mayoría de la lógica digital es binaria con dos estados (1 y ϕ).

DIGITIZE: "Digitalizar."
Proceso de conversión de una cantidad analógica en digital.

DIGITAL TO ANALOG CONVERTER: "Conversor digital a analógico."
Convierte la representación digital usada en los ordenadores en señales analógicas empleadas en el mundo real.

DIP (DUAL IN LINE PACKAGE): "Encapsulado de doble línea de pines."
Cápsula para los circuitos integrados normalizada, con 2 líneas de pines en paralelo.

DIRECT ACCESS: "Acceso directo."
Corresponde al proceso de obtener datos o colocar datos en memoria, donde el tiempo requerido por tal acceso es independiente de la localización del último dato obtenido o colocado en memoria.

DIRECT ADDRESING: "Direccionamiento directo."
Modo de direccionado normalizado, que se caracteriza por la facilidad en buscar directamente cualquier posición de una gran memoria.

DIRECT MEMORY ACCESS (DMA): "Acceso directo a memoria."
Método que permite almacenar y transferir datos con la memoria de un sistema sin participar la CPU en esta operación.

DIRECT MEMORY ACCESS CHANNEL: "Canal de acceso directo a memoria."
Método de I/O para un sistema que usa un procesador pequeño, cuya única labor es controlar la I/O. Con DMA los datos son introducidos o sacados del sistema sin la intervención de un programa.

DISABLE: "Desconectado, inhibido."
Proceso de inhibición de la función de un eemento.

DMA: "DMA."
Ver DIRECT MEMORY ACCESS.

DOS: "DOS."
Sistema operativo de disco.

DOT MATRIX: "Matríz de puntos."
Usada para identificar caracteres alfanuméricos.

DOUBLE PRECISION: "Doble precisión."
Corresponde al uso de dos palabras de ordenador para representar un número.

DUMP: "Vaciado o descarga."
Transferencia de la mayor parte del contenido de una sección de memoria a otra memoria. Copiar los contenidos de toda o parte de una memoria, normalmente desde un almacenamiento interno a un almacenamiento externo.

DUPLEX: "Duplex."
Método de operación de un circuito de comunicación, en el cual cada uno puede simultáneamente transmitir y recibir.

DYNAMIC MEMORY: "Memoria dinámica."
Elementos de memoria que almacenan datos, necesitando una continua acción de refresco de sus células para evitar su degradación. Cada bit es almacenado como carga de un condensador MOS. A causa de la descarga del condensador, cada célula debe ser refrescada cada 2 ms. mediante la reescritura de sus contenidos.

DYNAMIC STORAGE ELEMENTS: "Elementos de almacenamiento dinámico."
Elementos de almacenamiento, que contienen celdas, que deben ser refrescadas a intervalos de tiempo apropiados para prevenir la pérdida de la información contenida.

E

EAROM: "EAROM."
Memoria de sólo lectura, eléctricamente alterable. Se trata de una memoria ROM, cuyo contenido puede ser cambiado por procedimientos eléctricos.

EBCDIC: "Código EBCDIC."
Código de intercambio extendido del código decimal/binario. Es un código de 8 bits y 256 caracteres, usado para transmitir datos binarios.

ECHO: "Eco."
Acción de enviar un carácter de entrada desde un teclado hasta una impresora o display.

ECL CIRCUITS: "Circuitos ECL."
Circuitos de lógica bipolar de emisores múltiples.

EDGE TRIGGERING: "Disparo por flanco."
Activación de un circuito, cuando se produce el cambio de un impulso, o sea, en el flanco.

EDIT: "Revisar."
Modificar la forma o formato de datos.

ELECTROMAGNETIC INTERFERENCE (EMI): "Interferencia electromagnética."
Interferencia originada por campos eléctricos.

ELECTROSTATIC STORAGE: "Almacenamiento electrostático."
Dispositivo que almacena datos como áreas cargadas electrostáticamente en una superficie dieléctrica.

EMULATE: "Emular."
Imitar un sistema con otro, de tal forma que el sistema imitador acepte los mismos datos, ejecute los mismos programas, y obtenga los mismos resultados que el sistema imitado.

ENABLE: "Habilitar, permitir."
Señal de entrada que permite la función de un elemento.

ENCODE: "Codificar."
Aplicar un conjunto de reglas especificando la foma en que los datos deben ser representados, de tal forma, que luego puedan ser decodificados. Sinónimo de CODE.

END-AROUND CARRY: "Rotación de acarreo final."
Carry generado en la posición del dígito más significativo y enviado directamente al lugar del menos significativo.

ENTRY POINT: "Punto de entrada."
En una rutina, cualquier lugar al que se puede ceder el control.

EPROM: "EPROM."
"Memoria de sólo lectura de programa borrable." Una PROM que puede

ser vuelta a usar. La mayoría de las EPROM pueden ser borrables mediante exposición a rayos ultravioletas.

ERASE: "Borrar."
Borrar información de un medio de almacenamiento, por ejemplo, limpiar o escribir encima.

ERROR: "Error."
Cualquier discrepancia entre una cantidad computada, observada o medida y el verdadero valor o condición.

ERROR CORRECTING CODE: "Código de corrección de error."
Código que emplea bits extras para detectar y corregir errores.

EXCLUSIVE-OR FUNCTION: "Función OR exclusiva."
Forma modificada de la función OR, que tiene una ecuación lógica igual a la suma de cada elemento con el otro, estando ambos en estado opuesto en cada sumando.

$$A \text{ "EOR" } B = A\bar{B} + B\bar{A} = A \oplus B$$

EXECUTE: "Ejecutar."
Realizar una función específica o instrucción de computador. Arrancar un programa. Porción del ciclo de un computador durante el cual se lleva a cabo una palabra de control seleccionada o una instrucción.

EXECUTE CYCLE: "Ciclo de ejecución."
Ultimo ciclo de la realización de una instrucción. Durante este tiempo, se realiza la operación de la instrucción.

EXECUTION TIME: "Tiempo de ejecución."
Tiempo requerido para la ejecución de una instrucción.

EXPONENT: "Exponente."
Potencia de 10 por la que se multiplica un número. Se usa en la representación de coma flotante. Por ejemplo, el exponente del número decimal $0{,}9873 \times 10^7$ es el 7.

F

FALLING EDGE: "Flanco descendente."
Transición lógica de alto a bajo.

FAN-IN: "Abanico de entrada."
Máximo número de entradas que pueden conectarse a un circuito lógico.

FAN-OUT: "Abanico de salida."
Carga eléctrica que puede soportar una salida. En general, expresa el número de cargas entradas que puede excitar en paralelo la salida de un elemento lógico.

FEEDBACK: "Realimentación."
Información que, procedente de una o más salidas de un circuito, se aplica a las entradas para formar un bucle de control.

FEEDBACK LOOP: "Bucle de realimentación."
Componentes y procesos involucrados en la correción y control de un sistema, mediante el uso de parte de las entradas y salidas.

FETCH: "Búsqueda."
Fase de lectura de una instrucción de la memoria. Acción de obtener una instrucción de un programa almacenado y decodificar esa instrucción.

FIELD: "Campo."
En un registro, el área específica usada para una particular categoría de datos.

FIFO: "Primero en entrar, primero en salir."
Estructura de memoria en la que los datos se cargan de forma que el primero en entrar es el primero en salir. Una pila FIFO se usa para actuar de buffer (aislador) en la conexión de dos elementos que trabajan asíncronamente.

FIRMWARE: "Microprogramación cableada."
Programa almacenado en ROM.

FIXED POINT BINARY BUMBER: "Número binario de coma fija."

Número binario representado por un bit de signo y uno o más bits de magnitud, con una coma fija, situada entre dos bits seguidos.

FIXED-POINT REPRESENTATION: "Representación en coma fija."
Representación de números en los que la coma decimal se asigna a una posición fija.

FLAG: "Bandera o señalizador."
Bit de información que indica un estado, tal como la presencia de acarreo o sobrepasamiento de una operación.

FLIP-FLOP (storage element): "Biestable."
Circuito que tiene dos posibles estados estables y la capacidad de cambiar de uno a otro con la aplicación de una señal de control, manteniéndose en ese estado después de terminada dicha señal.

FLOATING: "Flotante."
Estado de alta impedancia o flotante en la salida de un elemento triestado.

FLOATING-POINT BINARY NUMBER: "Número binario de coma flotante."
Número binario expresado en forma exponencial. Esto es, una parte del número representa la mantisa y otra el exponente.

FLOATING-POINT REPRESENTATION: "Representación en coma flotante."
Técnica usada para representar una gran cantidad de números, usando una mantisa y un exponente. La precisión de la representación queda limitada por el número de bits que caben en la mantisa. La posición de la coma no tiene una posición fija.

FLOPPY DISK: "Discos flexibles o disquetes."
Elemento de memoria que usa discos flexibles para sostener la información.

FLOWCHART: "Diagrama de flujo."
Representación gráfica de la definición, análisis o solución de un problema, en el cual los símbolos se usan para representar operaciones, datos, flujos, equipos, etc.

FLOWCHART OR FLOW DIAGRAM: "Ordinograma o diagrama de flujo."
Ver FLOWCHART. Los ordinogramas permiten al diseñador visualizar el comportamiento de cada paso del programa.

FORMAT: "Formato."
Orden de los datos.

FORTRAN: "Sistema de traducción de fórmula."
Lenguaje usado principalmente para expresar programas de ordenador mediante fórmulas aritméticas.

FREE-RUN: "Marcha libre."
Proceso que permite a un circuito digital (típicamente un microprocesador) ponerse en marcha sin realimentación, proporcionando el estímulo a los restantes elementos del circuito.

FULL-ADDER: "Sumador completo."
Circuito lógico como el "Halfadder" semisumador, pero con la posibilidad de introducir el carry, procedente de una suma anterior.

FUNCTION: "Función."
Especifica el propósito de una entidad o misión característica.

G

GENERAL-PURPOSE COMPUTER: "Ordenador de propósito general."
Ordenador que está diseñado para manejar una gran variedad de problemas.

GLITCH: "Interferencia de baja frecuencia."
Impulso o tren de ruidos.

H

HALH-ADDER: "Semisumador."
Circuito lógico capaz de sumar dos números binarios, sin tener en cuenta

un posible acarreo de una suma anterior.

HALF SPLITTING: "División por mitades."
Técnica de reparación usada para resolver fallos de aislamiento.

HALT: "Paro."
Orden de parada del computador.

HAMMING CODE: "Código HAMMING."
Sistema de codificación de corrección de errores, usado en la transmisión de datos. Este código utiliza una matriz de control de paridad en su operación.

HAND ASSEMBLE: "Ensamblaje manual."
Tranducción de un programa en lenguaje ensamblador a código máquina, sin la asistencia de un programa Ensamblador.

HANDSHAKING: "Diálogo."
Proceso de consulta a los periféricos para saber si solicitan atención.

HARDWARE: "Circuitería."
Componentes individuales de un circuito, activos y pasivos. Hoy día, en lenguaje técnico, se da el nombre de hardware a cualquier elemento de los equipos de procesos de datos. En general es material físico, que contiene un ordenador u otros sistemas electrónicos.

HARD WIRED LOGIC: "Ver RANDOM LOGIC."

HAZARD: "Salida peligrosa no controlada."
Salida temporal de un circuito, que permite la salida de un valor no deseado durante el paso de un estado a otro.

HEXADECIMAL: "Hexadecimal."
Sistema de numeración de base 16. Los dígitos hexadecimales son: 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D y F.

HIGH-LEVEL LANGUAGE: "Lenguaje de alto nivel."
Lenguaje de programación orientado a aplicaciones, que se distingue del lenguaje máquina. Las instrucciones de un lenguaje de alto nivel no son directamente dependientes de la arquitectura de la máquina.

HIGH-ORDER: "Más peso o significado."
Bits de más significado que una palabra.

HOLD TIME: "Tiempo de sostenimiento."
El tiempo que el dato debe ser estable para permitir la realización y terminación de una señal de escritura.

I

INMEDIATE ADDRESSING: "Direccionamiento inmediato."
En este modo de direccionamiento el operando contiene el valor que interviene en la operación y no la dirección en que se encuentra.

IN-CIRCUIT EMULATOR: "Circuito emulador."
Ayuda de depuración que conecta al sistema bajo prueba, colocándolo en el zócalo del sistema microprocesador.

INCREMENT: "Incremento."
Adición de una unidad al contenido de un registro o una posición de memoria.

INDEX REGISTER: "Registro Indice."
Contiene la dirección base en el sistema de direccionamiento indexado. Es un registro cuyo contenido puede ser sumado o restado de la dirección del operando antes o durante la ejecución de una instrucción de ordenador.

INDEXED ADDRESSING: "Direccionamiento Indexado."
Modo por el que la dirección corriente se obtiene por un desplazamiento u offset, que se añade a la dirección base. La dirección se modifica por el contenido de un registro Indice, antes o durante la ejecución de una instrucción de ordenador.

INDEXING: "Indexado."
En ordenadores, método de modificar las direcciones que se ejecutan por medio de registro Indice.

INDIRECT ADDRESSING: "Direccionamiento indirecto."
Modo de direccionamiento que direcciona una posición de memoria, que contiene la dirección del dato que se ha de manejar.

INITIALIZATION: "Inicialización."
Coloca a un sistema en un estado conocido.

INPUT: "Entrada."
Señales introducidas al ordenador, circuito, o sistema, desde el mundo exterior.

INPUT/OUTPUT: "Entrada/salida."
Líneas o elementos usados para transferir con el exterior información del sistema.

INPUT PORT: "Puerta de entrada."
Circuito que conecta señales desde el exterior, como entradas, al sistema microprocesador.

INSTRUCTION: "Instrucción".
Orden simple de un programa. Las instrucciones pueden ser, lógicas, aritméticas, de memoria, de registros de entrada y salida, etc. Una secuencia de instrucciones constituye un programa.

INSTRUCTION COUNTER: "Contador de instrucciones."
Contador que indica la posición de la siguiente instrucción que ha de ser ejecutada.

INSTRUCTION CYCLE: "Ciclo de instrucción."
Conjunto de estados de la máquina para completar la ejecución de una instrucción.

INSTRUCTION DECODER: "Decodificador de instrucciones."
Unidad que transforma el programa de instrucciones en señales de control para el resto del sistema.

INSTRUCTION REGISTER: "Registro de instrucción."
Registro de microprocesador que contiene el Código OP de la instrucción que va a ejecutarse.

INSTRUCTION SET: "Juego de instrucciones."
Grupo completo de instrucciones que puede ejecutar un microprocesador dado.

INTERFACE: "Interconexión, adaptación."
Indica un circuito límite entre dos componentes, circuitos o sistemas adyacentes, que permite el intercambio de información entre ellos. También se usa para denominar al circuito que permite al microprocesador comunicarse con un periférico.

INTERLEAVE (o INTERLACE): "Intercalar."
Asignar números sucesivos de localizaciones de memoria separadas físicamente. Esto sirve para reducir el tiempo de acceso.

INTERRUPT: "Interrupción."
Supone la suspensión del normal desarrollo del programa que realiza el microprocesador, para poder atender a una solicitud de un servicio (interrupción). El procesador entonces salta desde el programa que está ejecutando a una rutina de servicio de interrupción. Cuando finaliza la rutina del servicio de interrupción, retorna el control al programa principal. Una interrupción implica la suspensión de una rutina normal de programa de un microprocesador en orden a tratar una demanda de servicio imprevista. La importancia de la capacidad de interrupción de un microprocesador depende de la clase de aplicaciones a las que será expuesto. Cuando un número de dispositivos periféricos interfieren un microprocesador, una o varias interrupciones simultáneas pueden ocurrir frecuentemente. Si se requieren múltiples demandas de interrupción, el procesador debe ser capaz de efectuar lo si-

guiente: demorar o impedir nuevas interrupciones, seleccionar una interrupción en orden a manejar la interrupción más urgente, establecer un método de manejo de interrupciones prioritarias, y, después de completar el servicio de interrupción, volver a empezar el programa interrumpido desde el punto en que fue interrumpido.

INTERRUPT MASK: "Máscara de interrupción."
Registro en el que cada bit controla una interrupción. Se usa para permitir, o no, ciertos tipos de interrupción.

INTERRUPT SERVICE ROUTINE: "Rutina de servicio de interrupción."
Programa que se ejecuta cuando sucede una interrupción. Similar a una subrutina, excepto que es inicializada por una señal física, denominada "Demanda de Interrupción".

INTERRUPT REQUEST: "Demanda de interrupción."
Señal lógica aplicada por la CPU, que inicia una rutina de servicio de interrupción.

INTERRUPT VECTORING: "Vector de interrupción."
Proporciona la dirección de salto a que se debe acudir cuando se reconoce una señal de interrupción. Permite que cada interrupción sea atendida por su adecuada rutina.

INTERVAL TIMER: "Contador o temporizador programable."
Se trata de un elemento programable, que permite controlar funciones de tiempo de contaje o de temporización.

I/O MAPPED: "Mapa de I/O."
Elementos de I/O a los que se accede mediante el uso de instrucciones, similares a las de tratamiento de memoria. Asigna a los elementos I/O diferentes posiciones de la memoria del sistema.

ITERATIVE: "Iterativo."
Proceso por el que se repite una serie de operaciones, hasta que se satisfaga una cierta condición. Usualmente queda implementado por un "loop", o bucle, en un programa.

J

JUMP: "Salto."
Instrucción que proporciona un cambio o rotura en la secuencia de un programa.

JUMP CONDITION: "Condición de salto."
Condiciones definidas en una tabla de transmisión, que determina los cambios de los flip-flops de un estado a otro.

K

K: "K."
Símbolo para $10^3 = 1.000$. Cuando se refiere a bits o palabras de memorias $K = 1.024$ (2^{10}).

KERNEL: "Núcleo central."
Mínimo circuito necesario para que funcione un microprocesador. Normalmente consta de CPU, clock, líneas de interrupción y DMA y alimentación.

KEYBOARD: "Teclado."
Conjunto de teclas usado para introducir información en los sistemas.

L

LABEL: "Etiqueta."
Nombre signado a una determinada posición de memoria. Cuando se usa el lenguaje Ensamblador, se hace referencia a posiciones de memoria referenciándolas por su etiqueta.

LANGUAGE: "Lenguaje."
Conjunto de representaciones, convenciones y reglas usadas para transmitir información.

LARGE SCALE INTEGRATION (LSI): "Alta escala de integración."
Tecnología que permite fabricar miles de semiconductores en un simple chip. La realización simultánea de gran cantidad de chips y la óptima densidad de agrupación de los componentes reduce el coste al maximizar el número de conexiones del sistema, hechas a nivel de chip. La complejidad del circuito permite sobrepasar las 100 puertas.

LATCH: "Báscula, cerrojo."
Elemento de hardware que captura información y la guarda (por ejemplo un grupo de flip-flops).

LEAST SIGNIFICANT BIT (LSB): "Bit de menos significado."
El bit situado más a la derecha de un número, que tiene el menor peso numérico o el valor más bajo. Los dígitos más a la derecha de un número.

LED: "Diodo emisor de luz."
Elemento semiconductor, que emite luz cuando es atravesado por una corriente.

LEVEL: "Nivel."
Grado de subordinación en una jerarquía.

LIFO: "Ultimo en entrar, primero en salir."
Ver stack.

LIGHT-PEN: "Pluma luminosa."
Pequeña célula fotoeléctrica en forma de una pluma, que va junto a la pantalla CRT para detectar el instante en que el rayo de electrones pasa a través de una localización particular durante su recorrido exploratorio.

LINEAR SELECT DECODING: "Técnica de decodificación de direcciones."
Emplea a los bits de direccionamiento más significativos para activar directamente los elementos del sistema.

LISTENER: "Escucha, oyente."
Elemento que acepta datos desde un bus de datos, como una puerta de salida.

LISTING: "Listado."
Salida de información de un programa de ordenador.

LOCATION: "Localización o posición."
Posición específica de almacenamiento en memoria, que está identificada por su dirección.

LOGIC ANALYZER: "Analizador lógico."
Sistema de prueba, capaz de visualizar 0 y 1, que constituye funciones complejas. Generalmente los analizadores lógicos disponen de 16 ó 32 líneas de entrada y pueden almacenar secuencias de 16 ó más bits sobre cada una de las líneas de entrada.

LOGIC COMPARATOR: Comparador lógico."
Instrumento de prueba, que compara el funcionamiento de un circuito integrado con el correcto conocido.

LOGIC PROBE: "Sonda lógica."
Herramienta de medida, que detecta el nivel lógico en los puntos de un circuito lógico.

LOGIC PULSER: "Generador lógico."
Herramienta de medida, que inyecta señales controladas en los puntos de interes de circuitos digitales.

LOGIC SHIFT: "Desplazamiento lógico."
Cambio que afecta a todas las posiciones.

LOGIC SIGNAL: "Señal lógica."
Señal de información, que transmite uno o dos posibles estados de significación.

LOOP: "Bucle."
Parte de un programa, que se repite al ejecutarse. Es una secuencia de instrucciones, que se ejecuta repetidas veces mientras prevalece una condición.

LOOP COUNTER: "Contador de bucle."
Registro, que guarda el número de veces que un loop (bucle) ha sido ejecutado.

LOWEST-ORDER: "Orden bajo."
Grupo de dígitos de un número, que tienen los pesos menores. Por ejemplo, los 3 bits de bajo orden del número 1110011101 son 101.

LSB: "Bit de menor significado."

LSI: "Ver LARGE SCALE INTEGRATION."

LSTTL: "LSTTL."
TTL Schottky de baja potencia. Circuitos digitales integrados, que emplean transistores Schottky para conseguir una relación velocidad/consumo superior a los TTL estandar.

M

MACHINE CODE: "Código máquina."
Ver MACHINE LANGUAJE. Código de operación que la máquina reconoce.

MACHINE CYCLE: "Ciclo máquina."
Período de tiempo básico para manipular datos en el sistema.

MACHINE LANGUAGE: "Lenguaje máquina."
Lenguaje binario, que es reconocido directamente por el procesador.

MACROINSTRUCTION: "Macroinstrucción."
Instrucción contenida en un lenguaje fuente, equivalente a una secuencia específica de instrucciones máquina.

MACROPROGRAMING: "Macroprogramación."
Programación con macroinstrucciones.

MAIN FRAME: "Procesador central."
Parecido a la unidad central de proceso.

MANTISSA: "Mantisa."
Valor fraccional, que se usa como parte de la representación de un número en coma flotante. Por ejemplo, la mantisa del número $0{,}9873 \times 10^7$ es 0,9873.

MASK: "Máscara."
Modelo de bits 1 y 0, que se emplean para seleccionar una función.

MASK PROGRAMMED: "Máscara programada."
Un circuito integrado, programado por una sola fotomáscara usada en la fabricación del chip.

MASS STORAGE: "Masa de memoria."
Memoria secundaria usada para almacenar gran cantidad de información. Generalmente a floppy-disk, discos o cassettes.

MATRIX: "Matriz."
En matemáticas, conjunto rectangular de 2 dimensiones que contiene cantidades. Las matrices se manejan de acuerdo con las reglas del álgebra de matrices. En ordenadores, red lógica en forma de un conjunto de líneas de entrada y salida, con elementos lógicos conectados a alguna de sus intersecciones. En memoria, disposición de las células básicas del conjunto.

MEDIUN SCALE INTEGRATION (MSI): "Escala media de integración."
Tecnología de construcción de circuitos integrados, que contienen una docena o más de funciones incluidas en el mismo chip.

MEMORY: "Memoria."
Parte de un sistema computador, en la que la información puede ser grabada y leída posteriormente. Normalmente sus tipos son de disco, de núcleos, de cinta, y de semiconductores (en las que se incluyen las RAM y las ROM). Término general que se refiere a cualquier forma de almacenamiento de información binaria. Los tipos de memoria básica funcional incluyen lectura/escritura y lectura solamente.

MEMORY MAP: "Mapa de memoria."
Muestra las direcciones asignadas a cada elemento del sistema.

MEMORY MAP I/O: "Mapa de memoria de I/O."

Los elementos de I/O son manejados por el mismo grupo de instrucciones y señales que si fuesen posiciones de memoria.

MICROCODE: "Microcódigo."
Ver Microprogram.

MICROCOMPUTER: "Microcomputador."
Sistema completo que incluye CPU, memoria e interfaces para I/O. La CPU suele estar formada por un microprocesador.

MICROPROCESSOR: "Microprocesador."
Unidad Central de Proceso (CPU) fabricada en un (a veces dos) chip. Consta de la ALU, el bloque de control y los registros de trabajo.

MICROPROGRAM: "Microprograma."
Programa que define el juego de instrucciones. El microprograma, también llamado microcódigo, ordena a la CPU lo que debe de hacer para ejecutar cada instrucción en lenguaje máquina. No es accesible al usuario.

MICROPROGRAMMING: "Microprogramación."
Técnica de control, usada para ejecutar la función de control del programa almacenado. Típicamente, la técnica consiste en usar un chip de memoria ROM preprogramada para contener varias secuencias de control, que normalmente ocurren juntas.

MNEMONIC: "Nemónico."
Instrucciones de ordenador escritas concisamente, (más fáciles para aprender) en forma de símbolos o palabras abreviadas. Por ejemplo ADD, SUB, CLR y MOV son códigos nemónicos para instrucciones, que serán ejecutadas como código máquina.

MNEMONIC CODE: "Código nemónico."
Códigos diseñados para ayudar a la memoria humana. El lenguaje del microprocesador es el código binario, que no es significativo para la mente humana. Para ser más accesible al hombre, a cada grupo de códigos binarios codificados se le asigna un grupo de letras, que guardan relación (abreviaturas) con el cometido de la instrucción.

MNEMONIC SYMBOL: "Símbolo nemónico."
Símbolo elegido para ayudar a la memoria humana; por ejemplo, una abreviatura tal como "MPY" para multiplicar.

MODEM: "Módem."
Modulador-demodulador juntos. Usado, por lo general, como interface de elementos digitales conectados vía teléfono. Codifica y decodifica series de bits en frecuencias.

MONITOR: "Monitor."
Programa que controla la operación de un microcomputador y permite al usuario realizar programas, examinar y modificar memoria, etc.

MOS: "MOS."
Metal-Oxido-Semiconductor. Circuitos integrados, hechos con transistores de efecto de campo.

MOST SIGNIFICANT BIT (MSB): "Bit de más significado."
Bit que ocupa la posición más a la izquierda de una palabra y que le corresponde el mayor peso.

MOTHERBOARD: "Placa base."
Ver Blackplane.

MPU: "MPU."
Unidad microprocesadora. Ver MICROPROCESSOR.

MSB: "Ver MOST SIGNIFICANT BIT."

MSI: "Ver MEDIUM SCALE INTEGRATION."

MTBF: "Tiempo medio entre averías o fallos."

MTTR: "Tiempo medio de reparación."

MULTIPLEX: "Multiplexar."
Transmitir simultáneamente 2 o más mensajes en un sólo canal.

MULTIPLEXING: "Multiplexado."
Proceso por el cual se transmiten varias señales por una misma línea.

Muy utilizado en sistemas de microprocesador en los que una línea es usada por diferentes informaciones o elementos en diferentes tiempos.

MULTIPLEXOR: "Multiplexor."
Dispositivo que permite enviar varias señales en una única transmisión. Normalmente, un multiplexor es similar a un interruptor con varias entradas y una sola salida.

N

NAK: "NAK."
Mensaje enviado a través de los enlaces de comunicación de datos para indicar que la recepción ha sido comprobada y, si es errónea, debe volver a transmitirse.

NEGATIVE LOGIC: "Lógica negativa."
El estado lógico falso (ϕ) se representa por el voltaje más positivo del sistema, y el estado verdadero o 1, por el más negativo. Para TTL el estado ϕ queda representado por un voltaje superior a 2,4 V y el estado 1 por menos de 0,4 V.

NESTED: "Encaje."
Subrutina, que es llamada desde otra subrutina o bucle mayor.

NODE: "Nudo."
Cualquier línea de señal conectada a dos o más elementos del circuito. Todas las entradas y salidas lógicas conectadas eléctricamente entre sí, son partes de un nudo.

NOISE: "Ruido."
Término que se refiere a señales eléctricas no deseadas o falsas.

NOISE INMUNITY: "Inmunidad al ruido."
Medida de la insensibilidad de un circuito ante las señales eléctricas falsas, o ante el ruido. El ruido puede ocurrir tanto en lógica positiva como negativa.

NON DESTRUCTIVE READ OUT: "Lectura exterior no destructiva."
Memoria diseñada de tal forma que la lectura no afecta a los contenidos almacenados. No es necesario ejecutar una escritura después de cada operación de lectura.

NUMERICAL CONTROL: "Control numérico."
Control automático de un proceso que se realiza por un dispositivo, el cual hace uso de todos, o parte, de los datos numéricos generalmente introducidos, cuando la operación se está realizando.

O

OBJECT CODE: "Código objeto."
Salida de un compilador o ensamblador que es ejecutable directamente por la máquina, pues está en código máquina, o está disponible para ser procesada y obtener un código máquina ejecutable.

OBJECT LENGUAJE: "Lenguaje objeto."
Lenguaje a que son traducidas las declaraciones, que son interpretados directamente por la máquina.

OBJECT PROGRAM: "Programa objeto."
Resultado final, después de traducir el lenguaje fuente (en ensamblador u otro lenguaje de alto nivel) a código máquina. El programa objeto está compuesto por instrucciones en código máquina, que el ordenador podrá ejecutar.

O.C.R.: "Reconocimiento de caracteres óptimos."

OCTAL: "Sistema Octal."
Sistema de numeración de base 8. Cada dígito octal corresponde directamente con tres dígitos binarios.

ONE'S COMPLEMENT: "Complemento a 1."
Sistema de representación de números en los que los números negativos se obtienen complementándolos. El bit más a la izquierda es el bit de signo, un cero para números positivos y un uno para los negativos.

OPCODE: "Código de operación." Ver OPERATION CODE.

OPEN-LOOP: "Bucle abierto."
Circuito que opera sin realimentación.

OPERAND: "Operando".
Dato sobre el que opera una instrucción.

OPERATING SYSTEM: "Sitema operativo."
Software que controla la ejecución de programas y que facilita informes, control input/output, contadores, compiladores, asimilación de memoria, manejo de datos, control de errores, servicios especiales.

OPERATION: "Operación."
Acción definida. El hecho de obtener un resultado a partir de uno o más operandos, de acuerdo con una regla que especifica el resultado para cualquier combinación de operandos.

OPERATION CODE: "Código de operación."
Parte de la instrucción en código máquina, que especifica la operación a realizar. Sinónimo de CODIGO DE INSTRUCCION.

OPTICAL CHARACTER RECOGNITION: "Reconocimiento de caracteres ópticos."
Identificación, por parte de la máquina, de caracteres impresos, usando para tal identificación dispositivos sensibles a la luz. Abreviadamente O.C.R.

OUTPUT-PORT: "Puerta de salida".
Circuito que permite al sistema microprocesador sacar señales a otros elementos.

OVERFLOW: "Sobrepasamiento."
Se produce cuando una operación aritmética genera una cantidad, que sobrepasa la capacidad de registro. Cuando esto ocurre, en el registro que contiene los flags, el correspondiente al sobrepasamiento se pone en 1.

P

PACK: "Paquete."
Condensar datos en un medio de almacenamiento, tomando como ventajoso el hecho de conocer las características de los datos, de tal forma que los datos originales puedan ser recuperados.

PAGE: "Página."
Normalmente, un conjunto de 256 direcciones de memoria. Los 8 bits de menos peso de una dirección especifican la posición dentro de la página, mientras que los 8 de más peso indican la página.

PARALLEL OPERATION: "Operación en paralelo."
Organización de la manipulación de datos dentro de la circuitería, donde todos los dígitos de una palabra se transmite simultáneamente en líneas separadas en orden a aumentar la velocidad de la operación.

PARAMETER: "Parámetro."
Variable que da un valor constante para un propósito o proceso específico.

PARITY: "Paridad."
Número de unos en una palabra, el cual puede ser par o impar. Cuando se usa la paridad, se emplea un bit extra para forzar al número de unos total a ser par o impar. Se emplean para detección de errores.

PARITY BIT: "Bit de paridad."
Bit de comprobación añadido a un conjunto de dígitos binarios para realizar la suma de todos los dígitos binarios incluido el bit de prueba, siempre par o siempre impar.

PARITY CHECK: "Prueba de paridad."
Técnica de añadir un bit a una palabra digital para calcular el total de unos o ceros binarios, siendo éstos unas veces par y otras impar. Este tipo de comprobación indicará que existe un error en los datos, pero no señalará la situación del error.

PATCH: "Remiendo."
Sección de codificación, insertada en una rutina para eliminar un error o alterarlo.

PC: PRINTED CIRCUIT: "Circuito impreso o PROGRAM COUNTER."
Contador de programa.

PCB: "Placa de circuito impreso."

PERIPHAL: "Periférico."
Cualquier elemento de exterior, conectado al computador.

PERIPHERAL EQUIPMENT: "Equipo periférico."
Unidades que trabajan en conjunción con un ordenador, pero que no forman parte de él.

P.L.A.: "Conjunto lógico programable."
Circuito integrado que emplea matrices ROM para combinar términos, sumas y productos de redes lógicas.

POINT TO: "Apuntar a."
Acto de especificación de una localización particular de memoria o registro.

POLLING: "Escrutinio."
Método empleado para identificar la fuente que ha hecho una petición de interrupción. La CPU escruta o lee los elementos para conocer el que ha pedido interrupción.

POP: "Lo mismo que PULL."
Operación de lectura de una palabra del stack.

PORT: "Puerta."
Zona en que los elementos del I/O se conectan al computador.

POSITIVE LOGIC: "Lógica positiva."
Lógica en la cual el voltaje más positivo representa el estado "1", el menos positivo representa el valor "0".

POWER-UP RESET: "Conexión de la alimentación. Reset."
Proceso de inicialización, que determina las condiciones del sistema al conexionar por primera vez la alimentación.

PRIORITY: "Prioridad."
Número asignado a un acontecimiento o elemento, que determina el orden en que será atendido en caso de existir en ese momento más elementos que reclaman servicio simultáneamente.

PRIORITY INTERRUPT: "Interrupción prioritaria."
Designación dada al método de obtener algunos comandos para tener preferencia sobre otros, de tal forma que se pueda dar una condición de prioridad de una operación sobre otra.

PROBLEM ORIENTED LENGUAGE: "Lenguaje orientado al problema."
Lenguaje de programación diseñado para la expresión conveniente de una clase de problema específica.

PROCESSING THE DATA: "Procesamiento de datos."
Manipulación u operación sobre un dato, como direccionador de un programa de ordenador.

PROCESSOR: "Procesador."
Lo mismo que microprocesador.

PROGRAM: "Programa."
Procedimiento para resolver un problema en forma asequible a un computador. Frecuentemente se denomina software.

PROGRAM COUNTER (PC): "Contador de programa."
Registro de la CPU, que almacena la dirección del próximo byte del programa que será leído. Las bifurcaciones requieren la carga en el PC de la dirección del salto, en caso contrario, el PC se incrementa en un byte después de cada lectura.

PROGRAM STATUS WORD (PSW): "Palabra de estado de programa."
Registro de propósito especial dentro del procesador, que contiene el estado y bits de control.

PROGRAMMABLE READ ONLY MEMORY (PROM): "Memoria de Lectura programable."

Tipo de memoria semiconductor, que contiene un programa fijo y que sólo puede ser leído. Puede ser programado después de haber creado el programa mediante un aparato llamado "Programador".

PROGRAMMING LENGUAGE: "Lenguaje de programación."
Lenguaje usado para escribir un programa. Puede ser máquina, ensamblador o de alto nivel.

PROM: "PROM."
Memoria de solo lectura programable. Ver: Programable Read Only Memory.

PROPAGATION DELAY: "Retardo de propagación."
Tiempo requerido por una señal para propagarse a través de un elemento.

PROTOCOL: "Protocolo."
Juego de reglas para intercambio de información.

PSEUDO INSTRUCTION: "Pseudo-instrucción."
Instrucción usada en el lenguaje Ensamblador. Las pseudo-instrucciones no tienen una correspondencia directa con el lenguaje máquina.

PSEUDO-OPERATION: "Pseudo-operación."
También denominada "Pseudo instrucción", "directivo ensamblador" o "pseudo-operación". Sirve para poder escribir una instrucción en lenguaje ensamblador.

PULL-UP RESISTOR: "Resistencia de polarización o de carga."
Resistencia empleada para proporcionar el suministro necesario de corriente a las puertas lógicas, triestado, con colector abierto.

PUSH: "Empujar."
Lo contrario a POP. Se trata de introducir en el stack una palabra.

PUSH-DOWN LIST: "Primero en entrar-primero en salir."
Lista construida y mantenida de tal forma que el elemento, que va a ser extraído, es el más antiguo de los que se encuentran todavía en la lista. (FIFO.)

PUSH-DOWN STACK: "Cargar el stack."
Ver STACK.

R

RADIX: "Base de numeración".
Número total de caracteres o números distintos, usado en un sistema de numeración.

RAM: "RAM."
Ver Random Access Memory. Memoria de lectura y escritura de acceso aleatorio.

RANDOM ACCESS: "Acceso aleatorio."
Método de acceso a memoria, que permite localizar cualquier palabra en cualquier tiempo y orden.

RANDOM ACCESS MEMORY: "Memoria de acceso aleatorio."
(RAM) Memoria de la que se puede obtener toda la información con, aproximadamente, el mismo tiempo de retardo, eligiendo una dirección al azar.

RANDOM LOGIC: "Lógica aleatoria."
El conexionado o alambrado de diseños lógicos requieren la interconexión de numerosos circuitos integrados, que representan elementos lógicos. La función del circuito se determina por los elementos funcionales o bloques funcionales y sus interconexiones por un programa.

READ ONLY MEMORY: (ROM) "Memoria de sólo lectura."
Elemento de memoria semiconductor con un programa fijo, que ha sido programado en la fábrica con un programa permanente.

REAL TIME: "Tiempo real."
Corresponde al tiempo actual durante el cual un proceso físico transcurre.

REDUNDANCY: "Redundancia."
Técnica consistente en usar más de un circuito del mismo tipo para implementar una función dada.

REFRESH: "Refresco."
Proceso de restauración de carga en las memorias dinámicas. Las RAM dinámicas deben ser recargadas todas sus células cada 2 ms.

REGISTER: "Registro."
Simple palabra de memoria. Los registros de la CPU son más fácilmente accesibles en su lectura que las posiciones de memoria externa. Los registros externos de la CPU son simplemente un grupo de flip-flop.

RELATIVE ADDRESSING: "Direccionamiento relativo."
Queda especificada la dirección del operando mediante un offset a la dirección normal, por ejemplo, tres bytes más arriba o más abajo.

RELATIVE ADRESS: "Dirección relativa."
Número que especifica la diferencia entre la dirección absoluta y la dirección base.

RESTORE: "Restauración."
Para retornar un registro u otra palabra del computador a su valor inicial o preestablecido.

RETURN: "Retorno."
En particular, se refiere a la instrucción, que al final de una subrutina proporciona el retorno a la dirección de la que se ha salido antes en el programa principal.

RETURN ADDRESS REGISTER: "Registro de dirección de retorno."
Registro empleado para guardar la dirección de la instrucción a ser ejecutada después de la ejecución de una subrutina.

RISING-EDGE: "Flanco ascendente."
Transición de bajo a alto.

ROM: "ROM."
Memoria de sólo lectura. Memoria con programa permanente.

S

SAMPLE-AND-HOLD-CIRCUIT: "Circuito de muestreo."
Circuito que ejecuta *la operación de muestrear* al nivel de voltaje analógico durante un corto período de tiempo y *almacenar* ese nivel de voltaje durante un período de tiempo mucho más largo, convirtiéndolo en digital.

SCANNING: "Barrido o rastreo."
Proceso de acceso secuencial a líneas independientes en un grupo determinado.

SCHMITT TRIGGER: "Circuito multivibrador de disparo."
Circuito con histéresis, usado para señales de entrada con ruidos o con tiempos de transición lentos.

SCRATCHPAD: "Memoria de apuntes."
Memoria que contiene datos intermedios necesarios para los resultados finales.

SELF-TEST: "Autoprueba."
Prueba de un producto realizada por sí mismo.

SEQUENCING: "Control secuencial."
Método de control usado para lograr una serie de pasos, que deben ocurrir en un orden particular.

SEQUENTIALL LOGIC: "Lógica secuencial."
Circuito cuyas salidas quedan determinadas por los estados previos (la historia) y el estado actual de las entradas. Compárese con lógica combinacional.

SEQUENTIAL LOGIC-SYSTEMS: "Sistemas de lógica secuencial."
Sistema digital que utiliza elementos de memoria.

SERIAL: "Serie."
Transmisión de bits de datos uno cada vez por la misma línea, en vez de emplear una línea para cada bit.

SERIAL ACCUMULATOR: "Acumulador de series."
Registro que recibe bits en serie o se-

cuencia y que, temporalmente, mantiene el dato para uso futuro.

SERIAL OPERATION: "Operación serie."
La organización de la manipulación de los datos dentro de la circuitería, donde los dígitos de una palabra son transmitidos uno cada vez a lo largo de una línea simple. El modo de operación en serie es más lento que el paralelo, pero utiliza menos circuitería.

SET-UP TIME: "Tiempo de preparación."
Tiempo que debe ser estable el dato antes de una señal de escritura. Mínima cantidad de tiempo que los datos deben estar presentes en una entrada para asegurar la aceptación de los datos, cuando el dispositivo está conectado.

SHIFT: "Desplazamiento."
Movimiento de caracteres de una unidad de información hacia la derecha o hacia la izquierda. Para un número binario esto es equivalente a irlo multiplicando o dividiendo por 2. Movimiento de datos hacia la derecha o hacia la izquierda.

SHIFT REGISTER: "Registro desplazamiento."
Registro en el cual el dato almacenado puede ser movido hacia la derecha o hacia la izquierda.

SIGN AND MAGNITUD NOTATION: "Notación de signo y magnitud."
Sistema de notación donde los números binarios se representan por medio de un bit de signo y uno o más bits de valor.

SIGNATURE: "Firma."
Valor de 4 dígitos generado por un analizador de señal, que se usa para caracterizar la actividad del dato presente en nudo durante un período de tiempo determinado.

SIGNATURE ANALYSIS: "Análisis de firma."
Técnica usada para la reparación de circuitos digitales. Se estimulan mediante señales los circuitos y se comparan las señales en los nudos con las correctas.

SIGNATURE ANALYZER: "Analizador de firmas."
Instrumento usado para convertir un flujo de datos en serie presentes en el microprocesador, en cuatro dígitos de señal.

SILO MEMORY: "Memoria SILO."
Leer datos almacenados de modo "First-in/first-out", el primero que llega es el primero que sale. También conocido como FIFO.

SIMULATE: "Simular."
Representar ciertas características del comportamiento de un sistema físico o abstracto mediante el comportamiento de otro sistema.

SIMULATOR: "Simulador."
Programa especial que simula la operación lógica del microprocesador. Permite que programas para un microprocesador puedan ser depurados en un sistema que emplea otro microprocesador. Dispositivo sistema, o programa de ordenador que representa ciertas características del comportamiento de un sistema físico o abstracto.

SINGLE-STEP: "Paso simple, o, paso a paso."
Proceso por el que se ejecuta un programa a base de una instrucción de ciclo máquina cada vez.

SINK CURRENT: "Corriente de carga."
Capacidad de entrada de corriente de un dispositivo.

SKIP: "Brinco."
Ignorar una o más instrucciones en una secuencia de instrucciones.

SMALL SCALE INTEGRATION (SSI): "Pequeña escala de integración."
Tecnología de menor complejidad que la MSI. Usualmente sus circuitos integrados contienen menos de 10 funciones.

SOFTWARE: "Software."
Ver PROGRAMS. Conjunto de programas de ordenador, procesos y do-

cumentación asociada, posiblemente relacionada con la operación de un sistema de proceso de datos. Por ejemplo, compiladores, rutinas de librería, manuales, diagramas de circuito.

SOLDER BRIDGE: "Puente de soldadura."
Gota por exceso de estaño en soldadura, que cortocircuita dos conductores. Problema usual en la fabricación de placas de circuito impreso. (PC).

SOURCE CODE: "Código fuente."
Programa escrito en lenguaje diferente al código máquina. Puede ser lenguaje ensamblador o de alto nivel. Programa codificado en un lenguaje, que no es lenguaje máquina (ni ensamblador o lenguaje compilador), que deberá ser traducido a lenguaje máquina para su empleo.

SOURCE CURRENT: "Fuente de corriente."
Capacidad de salida de corriente de un elemento.

SOURCE LANGUAGE: "Lenguaje fuente."
Lenguaje del que se traduce una respuesta.

SOURCE PROGRAM: "Programa fuente."
Programa de ordenador escrito en lenguaje fuente. Los programas en lenguaje compilador y ensamblador son lenguajes humanos, mientras que los programas objeto son lenguaje máquina.

SSI: "Ver SMAL SCALE INTEGRATION."

STACK: "Stack."
Bloque de sucesivas posiciones de memoria a las que se accede según el comportamiento de una pila LIFO. Para muchos microprocesadores, el stack puede ser cualquier bloque de sucesivas posiciones de la memoria de lectura/escritura de la memoria del sistema.

STACK POINTER: "Puntero de stack."
Es un registro de la CPU, que contiene la dirección de la cabeza del stack. En general, el puntero de stack se decrementa una unidad inmediatamente después de almacenar en el stack cada byte. Por el contrario, el puntero de stack se incrementa cuando se saca un dato del stack.

STATE: "Estado."
Condición de una entrada o salida de un circuito, tal que puede ser un "1" lógico o un "0" lógico. El estado de un circuito (puerta de flip-flop) se refiere a su salida. Un flip-flop se dice que está en el estado "1", cuando su salida Q = 1. Una puerta está enel estado "1", cuando su salida es "1".

STATIC MEMORY: "Memoria estática."
Elementos de memoria que no precisan de señales de reloj ni de refrescos.

STATIC STORAGE ELEMENTS: "Elementos de almacenamiento estático."
Elementos de almacenamiento que contienen celdas de almacenamiento para retener su información tanto tiempo como la tensión les es aplicada; a no ser que la información sea alterada por excitación externa.

STATUS: "Estado."
Condición presente de un elemento. Normalmente indicado por flags a base de flip-flop o por registros. Ver FLAG.

STORAGE: "Almacenamiento."
Ver MEMORY.

STORED PROGRAM: "Programa almacenado."
Conjunto de instrucciones en memoria, que especifica la operación que va a ser ejecutada.

STRESS TESTING: "Prueba de carga."
Introduciendo una carga mecánica, eléctrica o térmica, a un elemento eléctrico, modificar su funcionamiento y permitir observar problemas que puedan ocurrir.

SUBROUTINE: "Subrutina."
Parte de un programa que realiza una función bien definida. Se puede acceder a ella en diferentes partes del programa.

SOUBROUTINE CALL: "Llamada a subrutina."
El control del programa se transfiere a una subrutina, mientras se guarda la posición de la siguiente instrucción del programa principal a ser ejecutada en un RETRUN ADDRESS REGISTER (Registro de dirección de retorno).

SYNCHRONOUS CIRCUIT: "Circuito sincronizado."
Circuito en el cual todas las operaciones ordinarias son controladas por unas señales de reloj igualmente espaciadas.

SYSTEM: "Sistema."
Conjunto de métodos, procesos o técnicas, unidas por la interacción regular para formar un todo organizado.

T

TABLE: "Tabla."
Colección de datos ordenados para su lectura de forma cómoda. Frecuentemente, se almacenan en posiciones secuenciales de la memoria.

TABLE LOOK-UP: "Tabla de consulta."
Obtiene un valor desde una tabla de valores almacenados en el computador.

TELECOMMUNICATION: "Telecomunicaciones."
Perteneciente a la transmisión de señales a larga distancia, tal como a través del telégrafo, radio o televisión.

TEMPORARY STORAGE: "Almacenamiento temporal."
En programación, posiciones de almacenamiento reservadas para resultados intermedios.

TERMINAL: "Terminal."
Un punto en un sistema o red de comunicación en el cual los datos pueden entrar o salir.

THREE-STATE: "Triestado."
Elemento lógico cuya salida puede ser colocada en estado de alta impedancia, en adición a los otros dos estados usuales. Esta característica posibilita la unión en un mismo nudo de varias salidas de elementos triestado que es fundamental en el funcionamiento de los buses de datos de los microprocesadores.

THROUGHPUT: "Rendimiento."
Velocidad en la realización de problemas, de tratamiento de datos. El rendimiento varía de una aplicación a otra.

TIME DOMAIN: "Dominio del tiempo."
Información que es función directa del tiempo. La visualización de un osciloscopio es de dominio temporal.

TIMING PULSE: "Impulso de sincronismo."
Se emplea para control y sincronización de sucesos.

TRACER: "Indicador."
Ver CURRENT TRACER.

TRANSFER: "Transferencia."
Lo mismo que salto.

TRANSLATE: "Traducir."
Transformar sentencias de un lenguaje a otro sin variar significativamente el sentido (significado).

THROUBLESHOOT: "Reparación de averías."
Buscar un mal funcionamiento o fallo del programa para arreglar la avería.

TROUBLESHOOTING TREE: "Arbol de investigación de averías."
Diagrama de flujo que indica las pruebas y medidas a realizar para diagnosticar y localizar averías en un producto.

THRUTH TABLE: "Tabla de la verdad."

Diagrama que tabula y resume todas las combinaciones de posibles estados de las entradas y salidas de un circuito. Tabula lo que pasaría a la salida para una condición determinada de las entradas.

TTL: "TTL."
Lógica transistor-transistor. Familia de circuitos integrados digitales que tienen transistores bipolares de entrada y de salida.

TTY: "Teletipo."

TWO'S COMPLEMENT NUMBERS: "Números en complemento a dos." Sistema de numeración usado, por lo general, para representación de números positivos y negativos. Los números positivos en el sistema de complemento a 2 se representan igual que si fuesen números binarios. Sin embargo, los números negativos se representan en forma de complemento a 1 más 1 (Complemento a 2). El bit de más significado, si es un 0, indica número positivo y, si es 1, negativo.

U

UART: "Receptor-transmisor universal asíncrono."
Conversor serie-paralelo y paralelo-serie.

USASCII: "Código stándar para el intercambio de información (U.S.A.)." Es el código estándar usado en los EE. UU. para la transmisión de datos. Algunas veces se refiere a él como el "ASCII" código.

μC: "Microcomputador."

UNDIRECTIONAL: "Unidireccional." Cable o grupo de cables por los que la información circula en una sola dirección o sentido. Cada elemento conectado a un bus unidireccional puede ser transmisor o receptor, pero no ambos.

μP: "Microprocesador".

V

VARIABLE: "Variable."
Cantidad que puede tomar cualquiera de una serie de valores dados.

VECTOR INTERRUPT: "Ver interrupt vectoring."

VERY LARGE SCALE INTEGRATION (VLSI): "Muy alta escala de integración."
Tecnología por la que se fabrican en un solo chip cientos y miles de semiconductores.

VLSI: "Ver VERY LARGE SCALE INTEGRATION."

VOLATILE MEMORY: "Memoria volátil."
Elementos de memoria en los que cambian los datos almacenados, cuando se quita la alimentación. Las RAM pueden parecer como no volátiles, si se les acopla un dispositivo de protección, cuando se les desconecta la alimentación.

VOLATILE STORAGE: "Almacenamiento volátil."
Dispositivo de almacenamiento en el cual los datos almacenados se pierden cuando se elimina la corriente aplicada.

W

WALKING-ONES: "Balanceo de unos."
Memoria en la que un bit 1 es desplazado a través de cada posición de la memoria llenada con ceros. Un "balanceo de ceros" es lo contrario.

WORD: "Palabra."
Juego de caracteres que ocupan una posición de memoria y que el computador trata como una unidad. Ordinariamente, una palabra es tratada por la unidad de control como una instrucción y por la ALU como una cantidad.

WRITE: "Escribir."
Proceso de transferencia de información, usualmente, desde una memoria principal a un elemento de salida, para grabar datos en un registro, posición u otro elemento de almacenamiento.

WRITE ENABLE: "Permiso de escritura."
También llamado lectura/escritura o R&W. Señal de control de un elemento de almacenamiento o una memoria que activa la operación de escritura. Recíprocamente, cuando no está en modo de escritura, el modo de lectura está activado.

WRITE TIME: "Tiempo de escritura."
Tiempo durante el cual debe mantenerse el apropiado nivel en una línea de escritura y durante el cual los datos deben estar presentes para garantizar la escritura satisfactoria de los mismos en memoria.

OTROS LIBROS DE INTERES PUBLICADOS POR PARANINFO SA

Informática

ABRAMSON.– Teoría de la información y codificación.– 216 páginas.
APPELS y GEELS.– Técnica de conmutación con relés.– Biblioteca Técnica Philips.– 376 páginas.
ESCUDERO.– Reconocimiento de patrones.– 690 páginas.
FLORES.– Estructuración y proceso de datos. 480 páginas.
GARCIA MERAYO.– El lenguaje "Fortrán". Enseñanza programada.– 296 páginas.
GAUTHIER y PONTO.– Diseño de programas de sistemas.– 292 páginas.
GILDERSLEEVE.– Las tablas de decisión y su aplicación al proceso de datos. Enseñanza programada.– 208 páginas.
GILLE, DECAULNE y PELEGRIN.– Servosistemas. Teoría y cálculo.– 560 páginas.
GOLDSTEIN.– Programación lineal. Problemas y aplicaciones.– 576 páginas.
GOMEZ BARQUERO.– Fundamentos de servotecnia.– 250 páginas.
GRIES.– Construcción de compiladores.– 560 páginas.
HARTMAN, MATHES y PROEME.– Manual de los sistemas de información (ARDI).– 2 volúmenes: Vol. 1.– 384 páginas; Vol. 2. 384 páginas.
HENRION.– El ordenador y la mediana empresa.– 152 páginas.
HOERNES y HEILVEIL.– Introducción al álgebra de Boole y a los dispositivos lógicos. Enseñanza programada.– 352 páginas.
KASATKIN.– El ABC de la cibernética.– 152 páginas.
KEY.– Computación analógica para principiantes.– 156 páginas.
KORTHALS ALTES.– Circuitos transistorizados lógicos.– Biblioteca Técnica Philips.– 220 páginas.
LAMURE.– R.P.G. (GAP). Iniciación práctica.– 180 páginas.
LOTT.– Elementos del proceso de datos.– 310 páginas.
MOURSUND.– Cómo trabajan los ordenadores.– 164 páginas.
OLIVETTI.– Diccionario de informática inglés-español. Glosario de términos informáticos.– 256 páginas.
POLGAR.– Automatización y técnica del empleo de los relés.– 342 páginas.
SALTO DOLLA.– Diccionario de términos de proceso de datos inglés-español y español-inglés.– 356 páginas.
SANCHIS LLORCA y MORALES LOZANO.– Programación con el lenguaje PASCAL.– 356 páginas.
SASTRE FERNANDEZ.– Circuitos lógicos industriales y fuentes de alimentación estabilizadas con semiconductores.– 180 páginas.
SCHMIDT y MEYERS.– Introducción a los ordenadores y al proceso de datos.– 452 páginas.
SCHUH.– Principios de automatización.– 452 páginas.
TRAINING.– Introducción a la programación de los computadores.– 72 páginas.– Incluido en la Colección "Manuales Tecnológicos" publicada por Paraninfo.
VARIOS.– Cibernética. Aspectos y tendencias actuales.– 188 páginas.
VEEN.– Introducción a la teoría de la investigación operativa.– Biblioteca Técnica Philips.– 248 páginas.
WEBER.– Introducción a los métodos de la técnica digital.– AEG-Telefunken, Selecciones técnicas.– 196 páginas.
ZEINES.– Fundamentos de servomecanismos.– 290 páginas.

Electrónica

AEG-Telefunken.– Fundamentos de Electrónica de potencia.– 268 páginas.
AEG-Telefunken.– Técnica de control eléctrico y electrónico. Fundamentos, módulos y sistemas de representación práctica.– 240 páginas.
ANGULO.– Electrónica digital moderna.– 624 páginas.
ANGULO.– Electrónica fundamental.– 6 volúmenes:
Volúmen 1: 192 páginas.
Volumen 2: 224 páginas.
Volumen 3: 210 páginas
Volumen 4: 240 páginas.
Volumen 5: 260 páginas
Volumen 6: 352 páginas.
BERSTEIN.– Circuitos integrados C-MOS. Introducción, desarrollos y ejemplos de aplicación. Desarrollos y ejemplos de circuitos.– 296 páginas.
BESSON.– Amplificación de baja frecuencia. Teoría y práctica.– 492 páginas.
BILDSTEIN.– Filtros activos.– 292 páginas.
BORQUE.– Problemas de electrónica. Fórmulas y Tablas.– 144 páginas.
BUSCHER.– Charlas sobre electrónica.– 388 páginas.
CARLSON.– Reparaciones electrónicas en 10 minutos.– 176 páginas.
CARTER y DONKER.– Optoelectrónica aplicada. Teoría y práctica.
CASSIGNOL.– Física y electrónica de los semiconductores.– 342 páginas.
CASSIGNOL.– Teoría y práctica de los circuitos con semiconductores.– Electrónica lineal.– 360 páginas.
CASSIGNOL.– Teoría y práctica de los circuitos con semiconductores. Electrónica no lineal.– 288 páginas.
CHARIN.– Electrónica. Iniciación al cálculo y a la experimentación.– 2 volúmenes: Vol. 1.– 194 páginas; Vol. 2.– 208 páginas.
CLIFFORD.– Manual de datos electrónicos.– 184 páginas.
CLOSA.– Tecnología electrónica.– 2º curso.– 272 páginas.
DAMAYE.– Optoelectrónica.– 252 páginas.
D'AZZO y HOUPIS.– Sistemas lineales de control. Análisis y diseño convencional y moderno.– 752 páginas.
D'AZZO y HOUPIS.– Sistemas realimentados de control. Análisis y síntesis. Tratado moderno de servosistemas y automatización.– 804 páginas.
DIJK.– Introducción a la física electrónica.– 408 páginas.
DOKTER y STEINHAUER.– Electrónica digital. Fundamentos teóricos y técnica de circuitos.– 404 páginas.

1250

FIGHIERA.– Montajes electrónicos de iniciación.– 136 páginas.
FLORES y CLOSA.– Prácticas de electrónica. 1er curso.– 80 páginas.
FLORES y CLOSA.– Tecnología electrónica. 1er curso.- 242 páginas.
GARCIA BERLANGA.– Electrónica general. Enseñanza programada.– 456 páginas.
GARCIA y GUTIERREZ.– Empleo de los amplificadores operacionales.– 140 páginas.
GAUDRY.– Rectificadores, tiristores y triacs.– 350 páginas.
GLAS.– Electrónica con experimentos.– Biblioteca Técnica Philips.– 2 volúmenes: Vol. 1.– 304 páginas; Vol. 2.– 336 páginas.
GOMEZ DE TEJADA.– Dispositivos electrónicos básicos. Circuitos y tecnología.– 240 páginas.
GOMEZ DE TEJADA.– Electrónica básica. Tecnología.– 400 páginas.
GOMEZ DE TEJADA.– Tecnología electrónica.– 360 páginas.
GONZALEZ BERNALDO DE QUIROS.– Ingeniería electrónica.– 464 páginas.
GUILLIEN.– Problemas de electrónica con soluciones.– 456 páginas.
GUTENMAKHER.– Tratamiento electrónico de la información.– 120 páginas.
HAAS.– Calculadoras numéricas electrónicas. Fundamentos y componentes.– 280 páginas.
HAAS.– Laboratorio de electrónica.– 292 páginas.
HELLINGS.– Radar. Principios generales y aplicaciones.– 204 páginas.
HENDRIK JANSEN.– Circuitos electrónicos en el automóvil. Alarma. Protección. Encendido. Montaje. Radio. Magnetófonos. Varios.– 112 páginas.
HETTERSCHEID.– Amplificadores de paso de banda transistorizados.– 378 páginas.
HETTERSCHEID.– Diseño de amplificadores de F.I. transistorizados. Biblioteca Técnica Philips.– 290 páginas.
HINLOPEN.– Electrónica aplicada al automóvil.– 366 páginas.
JORDAN y BALMAIN.– Ondas electromagnéticas y sistemas radiantes.– 824 páginas.
JULANDER.– Magnitudes electrónicas, válvulas y semiconductores.– Biblioteca Técnica Philips.– 294 páginas.
JUSTER.– 30 montajes electrónicos de alarma.– 136 páginas.
KAUFMANN, GRABOWSKI y THOUZERY.– Análisis de las redes eléctricas.– 216 páginas.
KRETZMANN y ANGULO.– Electrónica y automática aplicadas a la industria.– 676 páginas.
LETRAUBLON.– Música electrónica.– 368 páginas.
LYNN, PULA, RINGELMAN y TIMMEL.– Amplificadores magnéticos de autosaturación.– 304 páginas.
MARTIN MARTIN.– Física y química para electricidad y electrónica. 1er curso.– 256 páginas.
MARTIN MARTIN.– Física y química para electricidad y electrónica. 2º curso.– 368 páginas.
MOOIJWEER.– Técnica de las microondas.– 360 páginas.
MORNAND.– Esquemas de electrónica.– 2 volúmenes: Vol. 1.– 228 páginas; Vol. 2.– 224 páginas.
MORRIS.– Electrónica moderna. Fundamentos y sus aplicaciones a la electricidad.– 336 páginas.
MUIDERKRING.– Electrónic Tube Handbook. 412 páginas.
NEETESON.– Núcleos de ferrita.– 256 páginas.
NEWCOMB.– Síntesis de circuitos activos integrados.– 360 páginas.
NIXON.– Transformación de Laplace. Fundamentos y aplicaciones. Tablas y ejemplos.– 272 páginas.
O'NEAL.– Sistemas electrónicos de proceso de datos.– 436 páginas.
OTTE, SALVERDA y WILLIGEN.– Del electrón al superheterodino.– Biblioteca Técnica Philips.– 428 páginas.– 5ª edición modernizada por A. Borque.
POOLE y JACKSON.– Ingeniería de control eléctrico.– 276 páginas.
QUINET.– Circuitos de electrónica y de amplificadores. Teoría y práctica. 3 volúmenes:
Volumen 1: 384 páginas.
Volumen 2: 480 páginas.
Volumen 3: 288 páginas.
QUINET.– Manipulaciones y medidas electrónicas.– 416 páginas.
RCA.– Electrónica básica. Curso programado do de circuitos.– 168 páginas.
ROPARS.– El radar en el mar. Teoría y empleo.– 262 páginas.
SASTRE FERNANDEZ.– Circuitos lógicos industriales y fuentes de alimentación estabilizadas con semiconductores.– 174 páginas.
SIREDEY.– Electrónica.– 416 páginas.
SMITH y SOROKIN.– El Laser.– 490 páginas.
STANDARD ELECTRICA.– I.T.T. Telecomunication planning. (Obra escrita en inglés).– 716 páginas.
SWEARER.– Alarma y protección. Sistemas electrónicos. Instalación y mantenimiento.– 288 páginas.
TAB BOOKS.– 1001 Circuitos electrónicos prácticos.– 606 páginas.
THALMANN.– Electrónica y radioelectricidad. 3 volúmenes:
Volumen 1: 396 páginas.
Volumen 2: 348 páginas.
Volumen 3: 296 páginas.
TRACTON.– Lógica y teoría de los amplificadores operacionales y lineales.– 200 páginas.
TREMOLIERES.– Electrónica y medicina.– 344 páginas.
TROUP.– Láseres y Máseres.– 164 páginas.
WARD y STRUM.– El flujograma de señal en el análisis de sistemas lineales. Enseñanza programada. 164 páginas.
ZEINES.– Principios de electrónica industrial. 408 páginas.

MICROPROCESADORES Arquitectura, programación y desarrollo de sistemas

JOSE Mª ANGULO